Scribe Notes for *Algorithmic Number Theory*  
Class 12—June 3, 1998  
Scribes: Jeremy Rotter, Yizhong Wang, and Wen Wang

**Abstract**

In this class, we discuss the extended Chinese remainder theorem and prove the NP-completeness of the anti-Chinese remainder theorem (ACRT).

1 **Extended Chinese remainder theorem**

Consider the system of congruences,

\[
\begin{align*}
x &\equiv x_1 \pmod{m_1} \\
x &\equiv x_2 \pmod{m_2} \\
 &\vdots \\
x &\equiv x_k \pmod{m_k}
\end{align*}
\]

(\textit{System }S)

**Theorem 1.1 (Extended Chinese remainder theorem).** The system of congruences $S$ has a solution if and only if $x_i \equiv x_j \pmod{\gcd(m_i, m_j)}$ for all $1 \leq i, j \leq k$. Furthermore, the solution is unique modulo $\text{lcm}(m_1, m_2, \ldots, m_k)$.

**Example 1.2.**

\[
\begin{align*}
x &\equiv 4 \pmod{6} \quad (1) \\
x &\equiv 2 \pmod{4} \quad (2) \\
x &\equiv 7 \pmod{9} \quad (3)
\end{align*}
\]

Before solving these equations, we need to check whether the solution exists or not. Since 9 and 4 are relatively prime, we only have to show that $\gcd(4, 6) \mid (4 - 2)$ and $\gcd(6, 9) \mid (9 - 6)$. Clearly, these are both true, so the conditions of the theorem are satisfied, hence there exists a solution.

To find the solution, we start with equations (1) and (2). From equation (1) we know that there exists a $t$, such that

\[x = 4 + 6t.\]

Substituting this into equation (2), we have,

\[4 + 6t \equiv 2 \pmod{4},\]

which is equivalent to

\[6t \equiv 2 \pmod{4}.\]

Then

\[3t \equiv 1 \pmod{2},\]

so,

\[t \equiv 1 \pmod{2},\]
i.e., \( t = 1 + 2j \) for some \( j \in \mathbb{Z} \). So, \( x = 4 + 6 + 12j = 10 + 12j \), or,
\[
x \equiv 10 \pmod{12} \quad (4)
\]
Now look at (3) and (4). From (3) we know that
\[
x = 7 + 9t'
\]
and we can plug this into (4) to get
\[
x = 7 + 9t' \equiv 10 \pmod{12}.
\]
We can subtract 7 from both sides to get
\[
9t' \equiv 3 \pmod{12}
\]
and then we can divide both sides by 3, giving us
\[
3t' \equiv 1 \pmod{4}
\]
or, since 3 is its own inverse modulo 4,
\[
t' \equiv 3 \pmod{4}.
\]
Now we can say that \( t' = 3 + 4j' \) for some \( j' \), so \( x = 7 + 9(3 + 4j) = 7 + 27 + 36j = 34 + 36j \).
Hence,
\[
x \equiv 34 \pmod{36},
\]
which satisfies the equations.

2 Anti-Chinese remainder theorem

**Definition 2.1.** The *Anti-Chinese remainder theorem* (ACRT) is a decision problem defined as follows:

*Instance:* Set \( S = \{(x_1, m_1), (x_2, m_2), \ldots, (x_k, m_k)\} \) of pairs of integers.
*Question:* Is there an integer \( x \) such that \( x \not\equiv x_i \pmod{m_i} \) for all \( 1 \leq i \leq k \)?

While implementations of the Chinese remainder theorem can be performed in polynomial time, it turns out that the Anti-Chinese remainder theorem is NP-complete. To show this, we need a known NP-complete problem that can be reduced to ACRT. Here, we will use the well-known NP-complete problem, 3-Satisfiability.

**Definition 2.2.** A literal is a variable or its complement, e.g., \( y_i \) or \( \overline{y_i} \).

**Definition 2.3.** A clause is a set of literals.

**Definition 2.4.** A clause is *satisfied* if and only if it contains at least one true literal.  

\(^1\)This implies that the logical or operation is performed on the literals in the clause.
Definition 2.5. The 3-Satisfiability problem (3SAT) is a decision problem defined as follows:

**Instance:** Set $U = \{y_1, y_2, \ldots, y_k\}$ of variables and a set $C$ of clauses over $U$ such that each $c \in C$ has cardinality 3.

**Question:** Is there a satisfying truth assignment for $C$; that is, an assignment of true or false to each $y_i$ such that each clause contains one or more true literals?

Example 2.6. Let $U = \{y_1, y_2, y_3, y_4\}$ and let $C = \{\{\overline{y}_1, y_2, y_4\}, \{\overline{y}_1, y_3, y_4\}, \{y_2, \overline{y}_3, \overline{y}_4\}\}$. The equivalent boolean expression to $C$ is

$$(y_1 \lor \overline{y}_2 \lor y_4) \land (y_1 \lor y_3 \lor y_4) \land (y_2 \lor \overline{y}_3 \lor \overline{y}_4).$$

One of the several truth assignments that satisfies $C$ is

- $y_1 \rightarrow true$
- $y_2 \rightarrow true$
- $y_3 \rightarrow false$
- $y_4 \rightarrow false$.

Theorem 2.7. ACRT is NP-complete.

Proof. First, we must show that ACRT $\in$ NP. Then, we must show that for every $L \in$ NP, $L \leq^P_m$ ACRT.

- A nondeterministic algorithm for ACRT is given as follows:
  - First, pick an $x$
  - Then, check whether $x \neq x_i \pmod{m_i}$ for $1 \leq i \leq k$ and accept if so.

Since we can restrict our guess to $0 \leq x \leq \text{lcm}(m_1m_2 \cdots m_k)$, $\log(x)$ can be bounded by $\log(S)$, where $S$ is the input size. This is to say, we can do the check in polynomial time. Hence, ACRT $\in$ NP.

- Instead of showing that for every $L \in$ NP, $L \leq^P_m$ ACRT, it will suffice to show that 3SAT $\leq^P_m$ ACRT.

Let $U = \{y_1, y_2, \ldots, y_t\}$ and $F = \{c_1, c_2, \ldots, c_n\}$ be an instance of 3SAT, where

$$c_i = \{z^{i}_a, z^{i}_b, z^{i}_c\},$$

and

$$z^{i}_a \in \{y_{a_i}, \overline{y}_{a_i}\}, z^{i}_b \in \{y_{b_i}, \overline{y}_{b_i}\}, z^{i}_c \in \{y_{c_i}, \overline{y}_{c_i}\}.$$ 

Let $p_1, p_2, \ldots, p_t$ be the first $t$ primes. Since $p_t = O(t \log t)$, we can generate this list of primes in polynomial time. Define

$$a'_i = \begin{cases} 
0 & \text{if } z^{i}_{a_i} = y_{a_i} \\
1 & \text{if } z^{i}_{a_i} = \overline{y}_{a_i}
\end{cases}$$

\begin{align*}
b'_i &= \begin{cases} 
0 & \text{if } z^{i}_{b_i} = y_{b_i} \\
1 & \text{if } z^{i}_{b_i} = \overline{y_{b_i}} \end{cases} \\
c'_i &= \begin{cases} 
0 & \text{if } z^{i}_{c_i} = y_{c_i} \\
1 & \text{if } z^{i}_{c_i} = \overline{y_{c_i}} \end{cases}
\end{align*}

**Example 2.8.** Below are the values of these variables for the set of clauses in Example 2.6.

\begin{align*}
a'_1 &= 0 \quad a'_2 = 1 \quad a'_3 = 0 \\
b'_1 &= 1 \quad b'_2 = 0 \quad b'_3 = 1 \\
c'_1 &= 0 \quad c'_2 = 1 \quad c'_3 = 1
\end{align*}

For $1 \leq i \leq n$, we can use the Chinese Remainder theorem to find an $x_i$ with $0 \leq x_i \leq p_{a_i}p_{b_i}p_{c_i}$, satisfying

\begin{align*}
x_i &\equiv a'_i \pmod{p_{a_i}} \\
x_i &\equiv b'_i \pmod{p_{b_i}} \\
x_i &\equiv c'_i \pmod{p_{c_i}}
\end{align*}

**Example 2.9.** Now, for Example 2.6, using Example 2.8, we can get the congruences

\begin{align*}
x_1 &\equiv 0 \pmod{2} \\
x_1 &\equiv 1 \pmod{3} \\
x_1 &\equiv 0 \pmod{7},
\end{align*}

and hence, $x_1$ can be uniquely determined modulo 42.

We can now define $S$ the following system of incongruences:

\begin{align*}
(1) &\quad \begin{cases} 
x \not\equiv 2 \pmod{3} \\
x \not\equiv 2, 3, 4 \pmod{5} \\
\vdots \\
x \not\equiv 2, 3, \cdots, p_i - 1 \pmod{p_i}
\end{cases} \quad \text{O}(t^3) \text{ incongruences} \\
(2) &\quad \begin{cases} 
x \not\equiv x_1 \pmod{p_{a_1}p_{b_1}p_{c_1}} \\
x \not\equiv x_2 \pmod{p_{a_2}p_{b_2}p_{c_2}} \\
\vdots \\
x \not\equiv x_n \pmod{p_{a_n}p_{b_n}p_{c_n}}
\end{cases} \quad \text{O}(n) \text{ incongruences}
\end{align*}

Now we can prove that $F$ is satisfiable if and only if this system of incongruences ((1) and (2)) has a solution. (1) is needed to ensure that $x \equiv 0, 1 \pmod{p_i}$ for all $1 \leq i \leq t$. Now let there be an assignment

\[(y_1, y_2, \cdots, y_n) = (y'_1, y'_2, \cdots, y'_n).\]
Then clause $c_i$ is satisfied if and only if

$$(y'_{a_i}, y'_{b_i}, y'_{c_i}) \neq (a'_{i}, b'_{i}, c'_{i}),$$

which by our construction means $x \neq x_i \pmod{p_{a_i}p_{b_i}p_{c_i}}$. Hence $3\text{SAT} \leq_p \text{ACRT}$.

We conclude that ACRT is NP-complete. \qed

3 \textbf{Next Time}

Next time we will finish up Chapter 5.