Complexity of the (Effective) Chinese Remainder Theorem

This is an alternate derivation[1] of the time complexity of the effective Chinese remainder theorem (CRT). The CRT allowed us to take a system of k congruences of the form $x \equiv a_i \pmod{n_i}$, where each of the n_i are pairwise co-prime, and find all the solutions, which are of the form:

$$x \equiv \sum_{i=1}^{k} a_i \frac{N}{n_i} \left[\left(\frac{N}{n_i} \right)^{-1} \right]_{n_i} \pmod{N} \tag{1}$$

where

$$N = \prod_{i=1}^{k} n_i.$$

This analysis proceeds by deriving time complexity bounds for the following operations:

1. Calculating N.
2. Calculating the additive terms in equation (1).
3. Calculating the sum of all these terms.

We must first calculate the value of N. For notational convenience, we define $N_i = \prod_{j=1}^{i} n_j$ and $\ell_i = \operatorname{len}(N_i)$. To calculate N_j requires that we have first calculated N_{j-1}, and then we multiply n_j with N_{j-1}. This single multiplication requires time $O(\operatorname{len}(n_j)\,\ell_{j-1})$. Note that $\operatorname{len}(n_j) = \operatorname{len}(N_j) - \operatorname{len}(N_{j-1}) = \ell_j - \ell_{j-1}$, so

$$\begin{aligned} \operatorname{len}(n_j)\,\ell_{j-1} &= (\ell_j - \ell_{j-1})\ell_{j-1} \\ &= \ell_j \ell_{j-1} - \ell_{j-1}^2 \\ &\leq \ell_j^2 - \ell_{j-1}^2. \end{aligned}$$

We can thus say that this single multiplication requires time $O(\ell_j^2 - \ell_{j-1}^2)$.

Calculating N_{j-1} in turn requires that we first calculate N_{j-2}, and so forth, down to N_2. ($N_1 = n_1$, so there is no calculation required for N_1). Summing, we find that calculating N_j can occur in $O(f(j))$, where

$$\begin{aligned} f(j) &= \sum_{i=2}^{j} \left(\ell_i^2 - \ell_{i-1}^2 \right) \qquad \textit{(A telescoping series!)} \\ &= (\cancel{\ell_2^2} - \ell_1^2) + (\cancel{\ell_3^2} - \cancel{\ell_2^2}) + \cdots + (\cancel{\ell_{j-1}^2} - \cancel{\ell_{j-2}^2}) + (\ell_j^2 - \cancel{\ell_{j-1}^2}) \\ &= \ell_j^2 - \ell_1^2. \end{aligned}$$

[1] The book asks you to develop a different approach in exercises 4.14 and 4.15, which were not assigned.

As $N = N_k$, calculating N can thus be accomplished in time

$$O(\text{len}(N)^2). \tag{2}$$

We now examine each of the terms of the sum in equation (1):

We do not need to calculate a_i (it is provided as input). The term a_i can be represented by a non-negative integer less than n_i, so it is of size no larger than $\text{len}(n_i)$.

The integer N/n_i has length no larger than $\text{len}(N) - \text{len}(n_i)$, so this division occurs in

$$O(\text{len}(n_i)(\text{len}(N) - \text{len}(n_i))). \tag{3}$$

The term $\left[(N/n_i)^{-1}\right]_{n_i}$ requires a few steps to calculate. The integer division was already calculated in the prior step, so we first calculate the integer $N/n_i \pmod{n_i}$, which requires another division (recall, division also provides us with the remainder!). The result of this division is no larger than $\text{len}(N) - 2\text{len}(n_i)$, so the division occurs in time

$$O(\text{len}(n_i)(\text{len}(N) - 2\text{len}(n_i))). \tag{4}$$

The remainder is no larger than $\text{len}(n_i)$. We then need to find the inverse of this remainder modulo n_i, which can occur using the extended euclidian algorithm; this result's length is again no longer than $\text{len}(n_i)$, and this inverse computation can occur in time

$$O(\text{len}(n_i)^2). \tag{5}$$

Multiplying the resulting a_i with N/n_i results in an integer less than N (as $a_i < n_i$), so the result is no larger than $\text{len}(N)$ and this multiplication computation occurs in time

$$O(\text{len}(n_i)(\text{len}(N) - \text{len}(n_i))). \tag{6}$$

Multiplying the above result with $\left[(N/n_i)^{-1}\right]_{n_i}$ results in an integer no larger than $\text{len}(N) + \text{len}(n_i)$, and this multiplication computation occurs in time

$$O(\text{len}(n_i)\,\text{len}(N)). \tag{7}$$

Reduction of this final product modulo N through division results in an integer no larger than $\text{len}(n_i)$, and this reduction computation occurs in time

$$O(\text{len}(n_i)\,\text{len}(N)). \tag{8}$$

Combining the results of equations (3), (4), (5), (6), (7), and (8), we find that term of the sum in equation (1) can be computed in time

$$
\begin{aligned}
&O(\mathrm{len}\,(n_i)\,(\mathrm{len}\,(N) - \mathrm{len}\,(n_i))) \\
+&O(\mathrm{len}\,(n_i)\,(\mathrm{len}\,(N) - 2\mathrm{len}\,(n_i))) \\
+&O(\mathrm{len}\,(n_i)^2) \\
+&O(\mathrm{len}\,(n_i)\,(\mathrm{len}\,(N) - \mathrm{len}\,(n_i))) \\
+&O(\mathrm{len}\,(n_i)\,\mathrm{len}\,(N)) \\
+&O(\mathrm{len}\,(n_i)\,\mathrm{len}\,(N)) \\
=&O(5\mathrm{len}\,(n_i)\,\mathrm{len}\,(N) - 3\mathrm{len}\,(n_i)^2) \\
=&O(\mathrm{len}\,(n_i)\,\mathrm{len}\,(N))
\end{aligned} \tag{9}
$$

Thus computing terms of the sum in equation (1) occurs in time $O(g(N))$ where

$$
\begin{aligned}
g(N) &= \sum_{i=1}^{k} \mathrm{len}\,(n_i)\,\mathrm{len}\,(N) \\
&= \mathrm{len}\,(N) \sum_{i=1}^{k} \mathrm{len}\,(n_i) && (10) \\
&= \mathrm{len}\,(N)^2 . && (11)
\end{aligned}
$$

The transition between equations (10) and (11) occurs because $N = \prod_{i=1}^{k} n_i$, so $\mathrm{len}\,(N) = \sum_{i=1}^{k} \mathrm{len}\,(n_i)$.

As previously noted, each of the terms in this sum are surprisingly of length no longer than $\mathrm{len}\,(n_i)$ after reduction, but we wish to do these additions modulo N, so a loose bound on this final summation computation would be $k - 1$ additions, each of which takes no more than $O(\mathrm{len}\,(N))$, which results in an integer result no longer than $\mathrm{len}\,(N)$, and this summation computation occurs in time $O((k-1)\mathrm{len}\,(N))$. We finally note that each $n_i \geq 2$, so $N > 2^k$, thus $\mathrm{len}\,(N) > k + 1 > k - 1$. Using this bound gives us a time complexity for the addition computation of

$$
O\left(\mathrm{len}\,(N)^2\right). \tag{12}
$$

Referring to equations (2), (11), and (12), we find that the entire effective CRT computation occurs in time $O(\mathrm{len}\,(N)^2)$.