Chinese Remainder Theorem

George E. Collins

October 12, 1998

The Chinesse remainder theorem states that if $m_1, m_2, \ldots, m_r$ are positive integers greater than or equal to 2 that are pairwise relatively prime, and if $0 \leq a_i < m_i$ for $1 \leq i \leq r$, then there is a unique integer A such that $0 \leq A < m = m_1 m_2 \cdots m_r$ and $A \equiv a_i \text{mod} m_1$ for $1 \leq i \leq r$. The most efficient algorithm for computing the integer a is one devised by H. L. Garner in 1958. SACLIB utilizes this algorithm, but only for the case $r = 2$ since repeated application of the $r = 2$ case suffices to solve the general case and is efficient in doing so. For example, if $r = 3$, we can first compute A_2 such that $A_2 < m_1 m_2$ and $A_2 \equiv a_i \text{mod} m_i$ for $i = 1$ and $i = 2$. Then we apply the algorithm again with $m_1 m_2$ and m_3 as moduli and with A_2 and a_3 as corresponding residues.

Also in SACLIB applications the m_i will always be odd β-integers. So the one Chinese remainder theorem algorithm that is needed is for the case of two moduli, m_1 and m_2, with corresponding residues a_1 and a_2. Let c be any integer such that $cm_1 \equiv 1(\text{modulo } m_2)$. Let $a = (a_2 - a_1)cm_1 + a_1$. Then $a \equiv a_1$modulo m_1 because $(a_2 - a_1)cm_1 \equiv 0(\text{modulo } m_1)$. Also, $cm_1 \equiv 1(\text{modulo } m_2)$, so $a \equiv (a_2 - a_1) + a_1 = 0(\text{modulo } m_2)$.

Computationally, we take c to be the inverse in Z_{m_2} of $H_{m_2}(m_1)$. We also replace a_1 by $a_1' = H_{m_2}(a_1)$ without affecting the validity of the above proof. After computing $(b = a_2 - a_1')c$ we replace b with $H_{m_2}(b)$ before multiplying by m_1. This also preserves the validity of the above proof. Then $0 \leq b \leq m_2 - 1$, so $0 \leq bm_1 + a_1 < m_1 m_2$, as required.

For the case that both moduli are small β-integers, these computations are performed by MDCRA. For the case that the first modulus is an arbitrary integer and the second is an odd β-integer, the SACLIB program MIDCRA is applicable. However, the specification of MIDCRA is slightly different. Its output A is required to be an element of Z'_{Mn}, where M and m are the two moduli. For any odd integer Z'_m is defined to be $\{a : |a| < m/2\}$.

In SACLIB the Chinese remainder theorem is always applied with each modulus m_i an odd β-integer prime. The integer c referred to above is input to MIDCRA instead of being computed by MIDCRA because typically the same pair of moduli will be used many times but with different residues. It is easy to verify that the computing time of MIDCRA for moduli M and m is codominant with $L(M)$.