Residue number systems

Intensive Computation

Annalisa Massini
2017/2018

Residue number systems

- Residue number systems are based on the *congruence* relation:
  - Two integers *a* and *b* are said to be ***congruent modulo m*** if *m* divides exactly the difference of *a* and *b*
  - We write $a \equiv b \pmod{m}$
- For example
  - $10 \equiv 7 \pmod{3}$
  - $10 \equiv 4 \pmod{3}$
  - $10 \equiv 1 \pmod{3}$
  - $10 \equiv -2 \pmod{3}$
- The number *m* is a *modulus* or *base*, and we assume that its values exclude 1, which produces only trivial congruences

Residue number systems

- In fact:
- If ***q*** and ***r*** are the **quotient** and **remainder**, respectively, of the integer division of a by m - that is: $a = q{:}m + r$

  → then, by definition, we have $a \equiv r \pmod{m}$

- The number ***r*** is said to be the ***residue* of *a* with respect to *m***, and we shall usually denote this by $r = |a|_m$

- The set of m smallest values, $\{0; 1; 2; \ldots; m - 1\}$, that the residue may assume is called the set of *least positive residues modulo m*

Residue number systems

- Suppose we have a set, $\{m_1; m_2; \ldots; m_N\}$, of N positive and pairwise **relatively prime** moduli
- Let M be the product of the moduli $M = m_1 \times m_2 \times \ldots \times m_N$

- We write the representation in the form $<x1; x2; \ldots; xN>$, where $xi = |X|_{mi}$, and we indicate the relationship between X and its residues by writing $X \approx <x1; x2; \ldots; xN>$

- Example: in the residue system {2, 3, 5}, M=30 and

$$8 \rightarrow <0, 2, 3>$$

$$16 \rightarrow <0, 1, 1>$$

Residue number systems

- Every number $X < M$ has a **unique representation** in the residue number system, which is the sequence of residues $<|X|_{mi} : 1 \leq i \leq N>$
- A partial proof of uniqueness is as follows:
  - Suppose X_1 and X_2 are two different numbers with the **same *residue representation***
  - Then $|X_1|_{mi} = |X_2|_{mi}$, and so $|X_1 - X_2|_{mi} = 0$
  - Therefore $X_1 - X_2$ is the least common multiple (**lcm**) of *mi*
  - But if the *mi* are relatively prime, then their **lcm** is *M*, and it must be that $X_1 - X_2$ is a multiple of *M*
  - So it cannot be that $X_1 < M$ and $X_2 < M$
  - Therefore, the representation $<|X|_{mi} : 1 \leq i \leq N>$ is unique and may be taken as the representation of *X*

Residue number systems

- The number M is called the *dynamic range* of the RNS, because the number of numbers that can be represented is M
- For unsigned numbers, that range is $[0;M - 1]$
- Representations in a system in which the moduli are not pairwise relatively prime will be not be unique: two or more numbers will have the same representation

- Example

	Relatively prime			Relatively non-prime		
N	**m1=2**	**m2=3**	**m3=5**	**m1=2**	**m2=4**	**m3=6**
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	0	2	2	0	2	2
3	1	0	3	1	3	3
4	0	1	4	0	0	4
5	1	2	0	1	1	5
6	0	0	1	0	2	0
7	1	1	2	1	3	1
8	0	2	3	0	0	2
9	1	0	4	1	1	3
10	0	1	0	0	2	4
11	1	2	1	1	3	5
12	0	0	2	0	0	0
13	1	1	3	1	1	1
14	0	2	4	0	2	2
15	1	0	0	1	3	3

Residue number systems

- We defined *standard residue number systems*
- There are also examples of *non-standard* RNS, the most common of which are the *redundant residue number systems*
- Such a system is obtained by, essentially, adding extra (redundant) moduli to a standard system
- The dynamic range then consists of a **legitimate** range, defined by the non-redundant moduli and an **illegitimate** range
- Redundant number systems of this type are especially useful in fault-tolerant computing

Residue number systems

- Ignoring other, more *practical*, issues, the best moduli are probably **prime numbers**

- For computer applications, it is important to have moduli-sets that facilitate both efficient representation and balance, meaning that the *differences between the moduli should be as small as possible*

Residue number systems

- Take, for example, the choice of 13 and 17 for the moduli that are adjacent prime numbers

- The dynamic range is 221

- With a straightforward binary encoding:
  - 4 bits will be required to represent 13
  - 5 bits will be required to represent 17

Residue number systems

- The representational efficiency is:
  - In the first case 13/16
  - In the second case is 17/32

- If instead we chose 13 and 16, then the representational efficiency:
  - is improved to 16/16 in the second case
  - but at the cost of **reduction in the range** (down to 208)
- With the better balanced pair, 15 and 16, we would have:
  - a better efficiency 15/16 and 16/16
  - A greater range: 240

Residue number systems

- It is also useful to have ***moduli that simplify*** the implementation of the ***arithmetic operations***

- This means that arithmetic on residue digits should ***not deviate too far from conventional arithmetic***, which is just **arithmetic modulo a power of two**

- A common choice of prime modulus that does not complicate arithmetic and which has good representational efficiency is $mi = 2^i - 1$

Residue number systems

- Not all pairs of numbers of the form $2^i - 1$ are relatively prime
- It can be shown that that 2^j - 1 and 2^k - 1 are relatively prime **if and only if** *j* and *k* are relatively prime

- For example:
  - 2^4-1= 15 15=3x5
  - 2^5-1= 31 31 *prime*
  - 2^6-1= 63 63=3x7
  - 2^7-1= 127 127 *prime*
  - 2^8-1= 255 255=3x5x17

Residue number systems

- Many moduli sets are based on these choices, but there are other possibilities; for example, moduli-sets of the form $\{2^n-1; 2^n; 2^n+1\}$ are among the most popular in use
- At least four considerations for the selection of moduli
  - The selected moduli must provide an **adequate range** whilst also ensuring that RNS representations are **unique**
  - The **efficiency of binary representations**; a balance between the different moduli in a given moduli-set is also important
  - The **implementations of arithmetic units** for RNS should to some extent be compatible with those for conventional arithmetic, especially given the legacy that exists for the latter
  - The **size of individual moduli**

Residue number systems

- One of the primary **advantages** of RNS is that certain RNS-arithmetic operations do not require carries between digits
- But, this is so only between *digits*
- Since a digit is ultimately represented in binary, there will be carries between bits, and therefore it is important to ensure that digits (→ the moduli) are **not too large**

Residue number systems

- Small digits make it possible to realize cost-effective table-lookup implementations of arithmetic operations
- But, on the other hand, if the moduli are small, then a large number of them may be required to ensure a sufficient dynamic range
- The choices depend on applications and technologies

Residue number systems

Negative numbers

- As with the conventional number systems, any one of the radix complement, diminished-radix complement, or sign-and-magnitude notations may be used in RNS
- The merits and drawbacks of choosing one over the other are similar to those for the conventional notations
- However, the **determination of sign** is much *more difficult* with the residue notations, as is **magnitude-comparison**
- This problem imposes many limitations on the application of RNS and we deal with just the positive numbers

Residue number systems

Basic arithmetic

- Addition/subtraction and multiplication are easily implemented with residue notation, depending on the choice of the moduli
- Division is much more difficult due to the difficulties of sign-determination and magnitude-comparison

Residue number systems

Basic arithmetic

- Residue **addition** is carried out by individually adding corresponding digits
- A **carry**-out from one digit position is **not propagated** into the next digit position
- As an example, with the moduli-set {2; 3; 5; 7}:
  - the representation of 17 is <1; 2; 2; 3>
  - the representation of 19 is <1; 1; 4; 5>
  - adding the two residue numbers yields <0; 0; 1; 1>, which is the representation for 36 in that system

Residue number systems

Basic arithmetic

- **Subtraction** may be carried out by negating (in whatever is the chosen notation) the subtrahend and adding to the minuend
- This is straightforward for numbers in diminished-radix complement or radix complement notation
- For sign-and-magnitude representation, a slight modification of the algorithm for conventional sign-and-magnitude is necessary:
  - the sign digit is fanned out to all positions
  - addition proceeds as in the case for unsigned numbers but with a conventional sign-and-magnitude algorithm.

Residue number systems

Basic arithmetic

- **Multiplication** too can be performed simply by multiplying corresponding residue digit-pairs, relative to the modulus for their position → multiply digits and ignore or adjust an appropriate part of the result
- As an example, with the moduli-set {2; 3; 5; 7}:
  - 17 → <1; 2; 2; 3>
  - 19 → <1; 1; 4; 5>
  - their product, 323 is <1; 2; 3; 1>

Residue number systems

Basic arithmetic

- Basic fixed-point division consists, essentially, of a sequence of subtractions, magnitude-comparisons, and selections of the quotient-digits
- But **comparison** in RNS is a diffcult operation, because RNS is not positional or weighted
- Example:
  - moduli-set {2; 3; 5; 7}
  - the number represented by <0; 0; 1; 1> is almost twice that represented by <1; 1; 4; 5>
  - but this is far from apparent

Residue number systems

Conversion

- The most direct way to convert from a conventional representation to a residue one is to divide by each of the given moduli and then collect the remainders, *forward conversion*
- This is a **costly** operation if the number is represented in an **arbitrary radix** and the **moduli are arbitrary**
- If number is represented in **radix-2** (or a radix that is a power of two) and the moduli are of a suitable form (e.g. 2^n-1), then these procedures that can be implemented with more efficiency

Residue number systems

Conversion

- The conversion from residue notation to a conventional notation - ***reverse conversion*** - is more difficult (conceptually, if not necessarily in the implementation) and so far has been one of the major impediments to the adoption use of RNS
  - One way in which it can be done is to assign weights to the digits of a residue representation and then produce a positional (weighted) mixed-radix representation that can then be converted into any conventional form
  - Another approach involves the use of the Chinese Remainder Theorem, which is the basis for many algorithms for conversion from residue to conventional notation