Intensive Computation – Homework 4

17 May 2021

Exercise 1 – Number representations

Write the **function** **`Radix10-to-RNS`** that:

- takes a natural number n and a radix-10 number X as **input**
- computes the dynamic range M for the module set < $m_1=2^n -1$; $m_2=2^n$; $m_3= 2^n+1$ >
- computes the RNS representation < x_1 x_2 x_3 > of X for the module set < $m_1=2^n -1$; $m_2=2^n$; $m_3= 2^n+1$ >
- gives as **output** the dynamic range M and the triple < x_1 x_2 x_3 >

Write the **function** **`Radix10-to-RB`** that:

- takes radix-10 number D as **input**
- computes the RB representation of D in canonical form as two binary strings N and R, one for normal bits and one for redundant bits
- gives as **output** the two binary strings N and R representing D

Write the **function** **`RB-to-Radix10`** that:

- takes two binary strings N and R representing an RB number as **input**
- computes the radix-10 associated value X
- gives as **output** either X or an error message if an overflow occurs (*with respect to the representation range corresponding to the number of bit considered)*

Write the **function** **`RB-Sum`** that:

- takes two RB numbers **A and B** as **input**
- computes the sum S applying the RB table
- gives as output S in RB representation

Write a script calling the functions above to compute the sum among a pairs of numbers randomly chosen in the range [0, M-1], where M is the dynamic range for the RNS representation with respect to a given n and module set < $m_1=2^n -1$; $m_2=2^n$; $m_3= 2^n+1$ >.
The script shows:

i) the values n and M and the two random values X1 and X2,
ii) the RNS representation of X1 and X2 and of the sum S=X1+X2,
iii) the strings obtained with the two steps of the RB sum for the three digits of the RNS representation.

Example n =4 M = 15x16x17 = 4080

X1 = 28 X2 = 35 R = X1+X2 = 63

X1 = < 13; 12; 11 > X2 = < 5; 3; 1 > R = X1+X2 = < 3; 15; 12 >

Exercise 2

Write the **functions `drawButterfly`** and **`drawBaseline`** that:

- takes the value n giving the size N=2^n of the network
- draw the Butterfly and the Baseline networks respectively

Write the **functions `self-routing-Butterfly`** and **`self-routing-Baseline`** that:

- takes a permutation in input
- calculates the routing on the Butterfly (Baseline respectively) using the self-routing algorithm
- gives as **output**:
  - the matrix P of size N x 2n, where each column represents a permutation, in particular:
    - ***odd*** columns represent the sequence ***before*** traversing nodes in a stage, and
    - ***even*** columns represent the sequence ***after*** traversing nodes in a stage;
    - assuming all inputs traverse the network in parallel, in the case a conflict occurs, the program stops in correspondence of that stage, after computing the switch setting for all the nodes in the current stage, keeping trace of the stage index where the conflict occurs
  - the matrix S of size (N/2) x n, representing the switch setting of the Butterfly, where 0 represents the state *straight*, 1 represents the state *cross*, NaN for switches where a conflict occurs and a value at your choice (any value different from 0, 1 and NaN) for switches not yet decided

Write the **script** that:

- takes the value n for expressing the size N=2^n of the Butterfly network as a parameter
- chooses a random permutation of N elements
- calls the function **`self-routing-Butterfly`** and **`self-routing-Baseline`**
- draws the picture of the Butterfly and the Baseline calling the function **`drawButterfly`** and **`drawButterfly`** using two different colors for the straight and cross state of nodes, and using the color **red** for nodes where a conflict occurs and the color **gray** for nodes of stages not traversed.

Remark The following definition of the Butterfly stage and Butterfly network can be useful:

An edge stage is a Butterfly edge stage *of size l, l= 1, 2, ⋯, n−1, if any node, with (n−1)-bit binary label v, in the first node stage is connected to node v′ in the second stage if and only if v = v′ (straight edge) or v and v′ differ only in the l-th bit (cross edge). A* Butterfly *is a log N stage network consisting of an ordered sequence of Butterfly edge stages of increasing size from 1 to n−1.*

Exercise 3 - Interconnection Networks – CLOS

Write a script (for the design of a Clos network) that:

- takes as input: **N**, number of inputs/outputs, and **n**, size of the crossbar module used for building switches – in the first stage, n is also the number of inputs allowed per module
- calculates the number of modules in each of the three stages of a Clos network, and the number of inputs and outputs of modules in both cases, ***strictly non-blocking*** and ***rearrangeable*** network
- writes in output the number and size of modules, for each of the three stages, in the two cases
- calculates the gain when using the Clos network, with respect to the Crossbar N x N, comparing their costs, in both cases strictly non-blocking and rearrangeable non-blocking,