ELEC 4511/5511

Lab 8

Lab 8 Vectorization of gradient calculation

In this lab, you will use Verilog to design a vectorized gradient calculation module, and verify your design using the provided testbench file.

1. Experiment goal:

- Be further familiar with hardware design using Verilog and Xilinx Vivado IDE
- Master the vectorization method of gradient calculation.
- Be further familiar with using testbench to do simulation.
  Understand the conversion method between matrix and vector in hardware programming

2. Background

For multivariate linear regression,

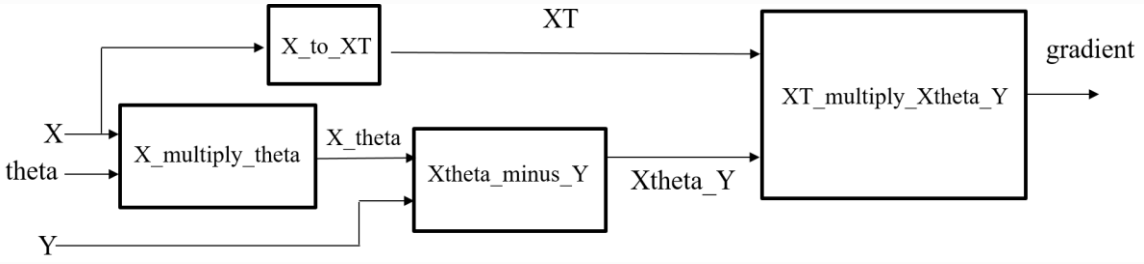
$$\hat{y}^{(i)} = \theta_0 + \theta_1 X_1^{(i)} + \theta_2 X_2^{(i)} + \cdots + \theta_n X_n^{(i)}$$

The gradient can be calculated by the following equations:

$$\bigtriangledown J(\theta) = \frac{2}{m}\begin{bmatrix} \sum_{i=1}^{m}(y^{(i)} - X^{(i)}\theta)\cdot(-X_0^{(i)}) \\ \sum_{i=1}^{m}(y^{(i)} - X^{(i)}\theta)\cdot(-X_1^{(i)}) \\ \sum_{i=1}^{m}(y^{(i)} - X^{(i)}\theta)\cdot(-X_2^{(i)}) \\ \cdots \\ \sum_{i=1}^{m}(y^{(i)} - X^{(i)}\theta)\cdot(-X_n^{(i)}) \end{bmatrix} = \frac{2}{m}\cdot X^T\cdot(X\theta - y)$$

$$X = \begin{bmatrix} 1 & X_1^{(1)} & X_2^{(1)} & \cdots & X_n^{(1)} \\ 1 & X_1^{(2)} & X_2^{(2)} & \cdots & X_n^{(2)} \\ \cdots & & & & \cdots \\ 1 & X_1^{(m)} & X_2^{(m)} & \cdots & X_n^{(m)} \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \cdots \\ \theta_n \end{bmatrix}$$

Matrix operations can be accelerated by hardware. An example design is shown below:



In this lab, you will design 4 sub-modules respectively, and then instantiate and connect them in the top-level module "gradient_calc".

3. Specific tasks:

1) Fully understand the provided Verilog file of the "**X_multiply_theta**" module.

```
module X_multiply_theta  #(parameter m =20, n= 3)
(
    input  [16*m*n-1:0] X,
    input  [16*n-1:0] theta,
    output reg  [32*m-1:0] X_theta
);

reg signed [15:0] X_tmp;
reg signed [15:0] theta_tmp;
reg signed [31:0] tmp;
integer i, j;

always@(*)begin

for (i=0;i<m;i=i+1)begin
    tmp = 0;
    for (j=0;j<n;j=j+1) begin
        X_tmp= X[16*(m-i)*n-1-16*j -:16];
        theta_tmp = theta[16*n-1-16*j -:16];
        tmp =  tmp + X_tmp* theta_tmp;
    end
    X_theta[32*m-1-32*i -: 32] = tmp;
end
end

endmodule
```

2) Follow the templates below to design the remaining three sub-modules. And test each module using the provided testbench file (**tb_Xtheta_minus_Y.v**, **tb_X_to_XT.v**, **tb_XT_multiply_Xtheta_Y.v**)

```
module Xtheta_minus_Y #(parameter m =20)
(
    input [32*m-1:0] X_theta,
    input [32*m-1:0] Y,
    output reg [32*m-1:0] Xtheta_Y
);


endmodule
```

```
module X_to_XT #(parameter m =20, n= 3)
(
    input  [16*m*n-1:0] X,
    output reg [16*n*m-1:0] XT
);


endmodule
```

```
module XT_multiply_Xtheta_Y  #(parameter m =20, n= 3)
(
    input  [16*n*m-1:0] XT,
    input  [32*m-1:0] Xtheta_Y,
    output reg  [32*n-1:0] gradient_vector
);



endmodule
```

3) Instantiate and connect these four sub-modules in the top-level module, "**gradient_calc**". And use the provided testbench file (**tb_gradient_calc.v**) to do the simulation.

Canvas submission: Please submit your report in ".**pdf**" file format and compress your Verilog code into a "**.zip**" file.

What needs to be included in your report:

1. Copy your all of the Verilog code in the report
2. Screenshot of the simulation result for each module
3. Please write a short analysis of each problem, mainly explaining how you think about the design of the function, what troubles you encountered during the design process, and how you finally solved them, etc.

What needs to be included in your "**.zip**" file:

Verilog code