PROJECT PROPOSAL
SMA5509

Topic: Implement FIR filter in parallel computing using Cilk

Name: **Pham Duc Minh**
Matric: HT030502H

Background

We have the discrete input signal is x(n), the output signal is y(n), the impulse response h(n). The output signal y(n) is the convolution of the input sequence x(n) and the filter h(n).

$$y(n) = \sum_m x(m)h(n-m)$$

or this formula can be written

$$y(n) = \sum_m h(m)x(n-m)$$

FIR filter

Discrete time LTI (Linearity and Time Invariance) systems can be classified into FIR and IIR.

An FIR filter has impulse response h(n) that extends only over a finite time interval, say $0 \leq n \leq M$, and is identically zero beyond that:

$\{h_0, h_1, h_2 \ldots, h_M, 0, 0, 0 \ldots\}$

M is referred to as the *filter order*.

The output y(n) in FIR is simplified to the finite-sum form:

$$y(n) = \sum_{m=0}^{M} h(m)x(n-m)$$

or, explicitly

$$y(n) = h_0x(n) + h_1x(n-1) + h_2x(n-2) + \ldots + h_Mx(n-M)$$

Methods to process

1. Block processing

a. Convolution

Noting the sum of the indices of h(m) and x(h-m) is m + (n-m) = n. Thus the convolution is

$$y(n) = \sum_{\substack{i,j \\ i+j=n}} x(m)h(n-m) \quad \text{(convolution table form)}$$

b. Direct Form

The length of the input signal x(n) is *L*.

The length (i.e, the number of filter coefficients) is

$$L_h = M + 1$$

The length of output y(n) will be

$$L_y = L_x + L_h - 1$$

m is in the range

$$\max(0, n - L + 1) \le m \le \min(n, M)$$

The direct form of convolution is given as follows:

$$y(n) = \sum_{\max(0,n-L+1)}^{\min(n,M)} h(m)x(n-m)$$

c. Matrix form

The convolution equations can also be written in the linear matrix form:

y=Hx

Matrix H is rectangular with dimensions

$$L_y \times L_x = (L + M) \times L$$

$$y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \dots \\ y_{L+M-1} \end{bmatrix} = \begin{bmatrix} h_0 & 0 & 0 & 0 & 0 \\ h_1 & h_0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & h_M \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_{L-1} \end{bmatrix} = \boldsymbol{Hx}$$

d. Overlap-Add Block Convolution Method

These above methods are not feasible in the application where the input is infinite or extremely long. A practical is to divide the long input into contiguous non-overlapping blocks of manageable length, say L samples, then filter each block and piece the output blocks together to obtain the overall output. Thus, processing is carried out block by block.

This is known as the overlap-add method of block convolution. Each of the input sub-blocks $\mathbf{x_0}$, $\mathbf{x_1}$, $\mathbf{x_2}$ . . . , is convolved with the order-M filter **h** producing the outputs blocks:

$\mathbf{y_0} = \mathbf{h} * \mathbf{x_0}$

$\mathbf{y_1} = \mathbf{h} * \mathbf{x_1}$

$\mathbf{y_2} = \mathbf{h} * \mathbf{x_2}$

and so on. The resulting blocks are pieced together according to their absolute timing. Block $\mathbf{y_0}$ starts at absolute time n = 0; block $\mathbf{y_1}$ starts at n = L because the corresponding input block $\mathbf{x_1}$ starts then; block $\mathbf{y_2}$ starts at n = 2L, and so forth. Because each output block is longer than the corresponding input block by M samples, the last M samples of each output block will overlap with the first M outputs of the next block. Note that only the next sub-block will be involved if we assume that 2L > L + M, or L > M. To get the correct output points, the overlapped portions must be added together (hence the name, overlap-add).

2. Sampling Processing

Convolution methods process the input signal on a block-by-block basis. Sampling processing operate on a sample-by-sample basis. This method is convenient of real-time applications that require the continuous processing of the incoming input.

The direct form I/O convolutional equation for an FIR filter of order M is given by

$$y(n) = h_0 x(n) + h_1 x(n-1) + ...h_M x(n-M)$$

We define the internal states $w_1(n)$, $w_2(2)$, $w_3(n)$….

$w_0(n) = x(n)$

$w_1(n) = x(n-1) = w_0(n-1)$

$w_2(n) = x(n-2) = w_1(n-1)$

…..

With this definition, we can y(n) in the form

$$y(n) = h_0 w_0(n) + h_1 w_1(n-1) + ...h_M w_M(n)$$

Goal of project

- ✓ Use Cilk to implement FIR filter in parallel. Try to shorten processing time of the filer.
- ✓ Compare with the processes of DSP kit (pipeline) and the process of FPGA (also parallel).

How to implement

- ✓ Multithreaded programming
- ✓ Cilk compiler

References

Introduction to Signal Processing, Sophocles J.Orfanidis, ISBN 0-13-209172-0.
A Minicourse on Multithreaded Programming, Charles E. Leiserson, Harald Prokop
Cilk 5.3.2 Reference Manual, MIT Laboratory for Computer Science