Phys 7411:
Fast Fourier Transform

Juana Moreno - moreno@lsu.edu
and
Karen Tomko – ktomko@osc.edu
Spring 2009

Fast Fourier Transform

- Extensively used in:
  - Time series and waveform analysis
  - Linear partial differential equations
  - Convolution
  - Digital signal processing
  - Image filtering, etc
- Reading (Grama, Introduction to Parallel Computer: Chapter 13 & Quinn, Parallel Programming in C with MPI and OpenMP: Chapter 15)

Fast Fourier Transform:

http://www.fftw.org

FFT algorithm:

Consider a sequence X={X[0],X[1]…X[n-1]}, the discrete Fourier transform Y={Y[0],Y[1]…Y[n-1]}:

$$Y[j] = \sum_{k=0}^{n-1} X[k]\,\omega^{kj},\ 0 \le j < n;\ \omega = e^{2\pi i/n}$$

Complexity of computing entire sequence= $O(n^2)$

FFT algorithm:

$$Y[j] = \sum_{k=0}^{(n/2)-1} X[2k]\,\omega^{2kj} + \sum_{k=0}^{(n/2)-1} X[2k+1]\,\omega^{(2k+1)j} =$$

$$= \sum_{k=0}^{(n/2)-1} X[2k]\,e^{2(2\pi i/n)kj} + \sum_{k=0}^{(n/2)-1} X[2k+1]\,\omega^{j}\,e^{2(2\pi i/n)kj} =$$

$$= \sum_{k=0}^{(n/2)-1} X[2k]\,e^{2\pi ikj/(n/2)} + \omega^{j}\sum_{k=0}^{(n/2)-1} X[2k+1]\;e^{2\pi ikj/(n/2)};$$

$$\varpi = e^{2\pi i/(n/2)} = \omega^{2}$$

$$Y[j] = \sum_{k=0}^{(n/2)-1} X[2k]\,\varpi^{kj} + \omega^{j}\sum_{k=0}^{(n/2)-1} X[2k+1]\;\varpi^{kj}$$

(n/2)-point computation → recursive algorithm

$$Y[i] = [X[0]\varpi^0 + X[2]\varpi^i + X[4]\varpi^{2i} + X[6]\varpi^{3i}] +$$
$$+ \omega^i [X[1]\varpi^0 + X[3]\varpi^i + X[5]\varpi^{2i} + X[7]\varpi^{3i}] =$$
$$= [X[0]\varpi^0 + X[4]\varpi^{2i}] + \varpi^i [X[2]\varpi^0 + X[6]\varpi^{2i}] +$$
$$+ \omega^i [X[1]\varpi^0 + X[5]\varpi^{2i}] + \omega^i \varpi^i [X[3]\varpi^0 + X[7]\varpi^{2i}] =$$
$$= [X[0]\omega^0 + X[4]\omega^{4i}] + \omega^{2i} [X[2]\omega^0 + X[6]\omega^{4i}] +$$
$$+ \omega^i [X[1]\omega^0 + X[5]\omega^{4i}] + \omega^{3i} [X[3]\omega^0 + X[7]\omega^{4i}] =$$
$$= [X[0]\omega^0 + X[4]\omega^{4i}] + [X[2]\omega^{2i} + X[6]\omega^{6i}] +$$
$$+ [X[1]\omega^i + X[5]\omega^{5i}] + [X[3]\omega^{3i} + X[7]\omega^{7i}]$$

Maximun number of levels of recursion is **log n**

At the mth level, 2^m FFTs of size $n/2^m$ are computed

Complexity **O(n log n)**