Fast Fourier Transform with BrookGPU

CS594 GPU Programming
Julian Yu-Chung Chen
2006-04-25
GPGPU

- Modern GPU is fast & programmable
- Out-performs CPU in some cases
Fast Fourier Transform

- Frequently used in signal processing, compression etc.
- Computation intensive
- Moreland & Angel’s GPU implementation
  - Cannot have branch in frag. program
  - Need multiple frag. program switch

\[
F(k) = \sum_{n=0}^{N-1} f(n) W_N^{kn},
\]

\[
W_N^{kn} = e^{-j\frac{2\pi kn}{N}} ; f(n), n = 0, \ldots, N - 1,
\]
Figure 48-1. The Decimation-in-Time Butterfly Algorithm

Used for computing the DFT of a discrete digital signal of eight samples.
Tangle + Untangle

• Perform FFT on $h(x) = f(x) + j g(x)$
• FFT is linear: $H(u) = F(u) + j G(u)$

$$F(u)_R = \frac{1}{2} \left( H(u)_R + H(N-u)_R \right)$$
$$F(u)_I = \frac{1}{2} \left( H(u)_I - H(N-u)_I \right)$$
$$G(u)_R = \frac{1}{2} \left( H(u)_I + H(N-u)_I \right)$$
$$G(u)_I = -\frac{1}{2} \left( H(u)_R - H(N-u)_R \right)$$
BrookGPU

- From Stanford Univ.
- Compiler & runtime of Brook stream programming language
- Ease GPGPU programming
- Research program, still beta
BrookGPU

- stream function: kernel
- stream datatype: float2, float4
- can do something like: streamSwap(s, s_out);
- Translate to embedded Cg program
- Rendering to offscreen pbuffer
  - Nice to have in cluster environment
Results & Issues

- export BRT_RUNTIME=[ogl|cpu]
- Brook’s CPU backend is way slow!
  - 0.5 sec vs. 1 min 3.7 sec on 512x512 input
- Should compared to state-of-art CPU FFT implementation: http://www.fftw.org/
- Program complains when input size larger than 1024x1024: cannot allocate pbuffer
GPU vs. CPU

Figure 3: Time to compute one FFT (latency), as a function of the size of the FFT. Toward the bottom of the graph is faster. “fftw” is the CPU-based FFTW; “ogl” is the GPU-based Brook OpenGL back end to libgpuft.
Results
Multiple GPUs?

- MPI + BrookGPU
- Can do bigger problem size
- Significant speedup?
- Meaningful for using GPUs?
Some thoughts

• Use nVidia cards + Cg!
• Beneficial under certain situations
• Graphics clusters
• Graphics resource monitoring
  • Available pbuffer size?
Issues

- GPU today: Single-precision floating-point
- Need IEEE-compliant, double, even 64-bit?
- Exceptions: eg. divide-by-zero
- Size limitation
- Programming model
References

- http://www.gpgpu.org/
- http://graphics.stanford.edu/projects/brookgpu/
- http://www.umiacs.umd.edu/research/GPU/
- http://libgpufft.sourceforge.net/
- http://graphics.stanford.edu/projects/gpubench/