Secure GPU Computing Over LAN

Attique Dawood

January 22, 2012

1 GPU Computing

Graphics processing units (GPU) have evolved over the last half a decade from a graphics–only processor to a general purpose computing device. General purpose GPU (GPGPU) computing is the use of GPUs for engineering and scientific computations. The two major manufacturers of GPUs are nVidia and AMD/ATi. Both offer tools and APIs to program their GPUs.

GPUs nowdays are capable of performing teraflops of computations. The GPGPU computing model is to have CPU work in conjunction with GPU. The computationally intensive part is run on GPU while the rest of the program executes on CPU. Figure 1 illustrates this.

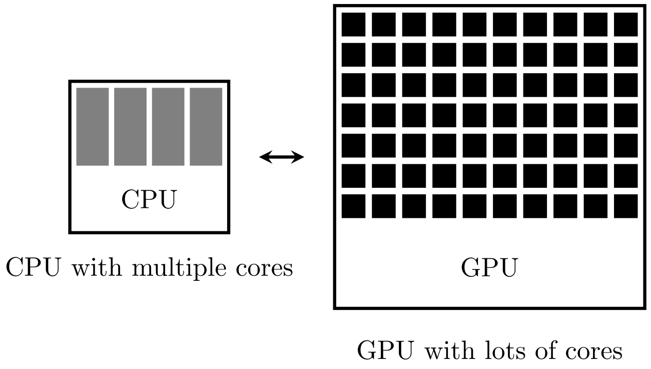

Figure 1: GPGPU Computing.

2 Project Overview

A central server will be connected to multiple computers (hosts) with GPUs (devices). Clients will connect to the server through a secure channel. Clients will request computational resource(s) and upload their code. The server will look for any free devices and make allocation as necessary. Client code will compile and run on a free device and output will be returned. In case of compilation errors client will be notified.

3 Project Phases

This project is divided into three phases:

3.1 Phase I: Network Programming

The first phase is to implement the client–server at application layer. Server should be able to handle multiple clients at a time. It should also be able to support multiple simulataneous connections both to clients and GPU hosts. Server will maintain a dynamic list of computational resources available and those which are in use.

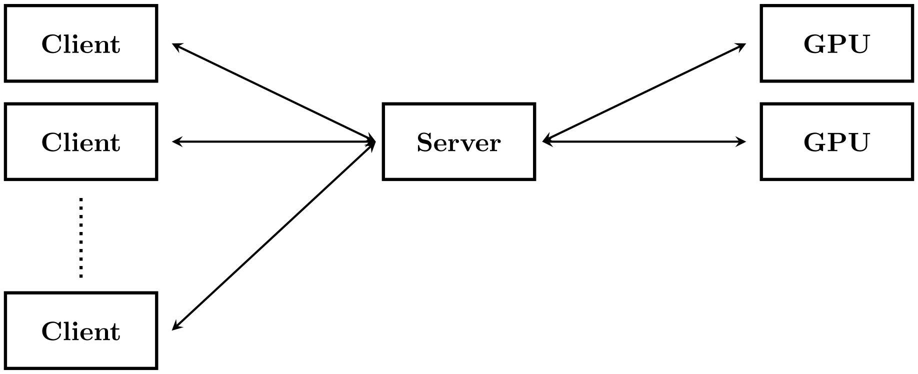

Figure 2: Project Overview.

3.2 Phase II: GPU Programming

The hosts with GPUs should be able to receive client code and compile it on the fly. In case of errors, appropriate error messages must be relayed to client through server. A mini–compiler can be implemented at the host which can convert conventional loop–based code into appropriate kernel code to be run on GPU.

Another aspect is multi–GPU computing where a number of GPUs are available to the client. Client can request multiple GPUs in order to speed up the computation. In hybrid multi–GPU context, multiple GPUs running different APIs such as CUDA and OPENCL can be combined to work on a single problem. Synchronization is an important issue in such a scenario.

3.3 Phase III: Security

All communication from clients to server to GPU hosts must be secure. Network traffic must be encrypted to keep out eavesdropping. PGP can be a good choice of security. Clients can be assigned usernames/passwords beforehand to only allow trusted access. PGP on top of it can provide additional layer of security if any password(s) are compromised.

4 Project Timeline

Major focus will be on the network programming part. Rest of the functionality will depend on robustness and features of client–server application.

Following table provides a tentative timeline for this project:

Week	**Duration**	**Tasks**
2	1 week	Set and install network programming APIs on PCs
3	1 week	Implement a simple client–server application for handling multiple clients
4-6	3 weeks	Implement core features of network programming
7-8	2 weeks	Set up GPUs and install GPU programming APIs
9-10	2 weeks	Provide GPU programming interface to clients through server over network
11-12	2 weeks	Refinements
13-15	3 weeks	Encrytion of network traffic and additional security features like PGP

Table 1: Project timeline.

References

[1] http://en.wikipedia.org/wiki/GPGPU

[2] http://www.nvidia.com/object/GPU_Computing.html

[3] http://www.amd.com/us/products/technologies/stream-technology/opencl/pages/gpgpu-history.aspx