Initial Plan : Parallelising the L-BFGS algorithm on GPUs for Quantum Control problems

AUTHOR: MAX CHANDLER
SUPERVISOR: DR FRANK LANGBEIN

Student Number:	C1125169
Module :	CM3203
Module Title:	One Semester Individual Project
Credits:	40

Project Description

Quantum control is the process of finding optimal ways to control and manipulate quantum systems. In this project, I will be focusing on systems modelled around the Schrödinger equation, which describes how the quantum state of a physical system changes over time. When solving quantum control problem there are several components needed: a description of the evolution of the system (System Hamiltonian), a description of the effect of the control (Control Hamiltonian) and the target operator which describes the overall required evolution of the system. The output will be a set of piecewise controls which indicates how strongly the control hamiltonian is applied over time. In this project, focus will be specifically on implementing an operator or gate over a fixed target time.

The aim of the project is to implement a framework which provides accurate results in less time when solving quantum control problems through the use of parallelisation techniques. One of the current algorithms for solving these problems is the Limited-Memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS), an iterative, non-linear optimisation algorithm. As it is an iterative process there is a limited amount of possible speedup. However, parallelising the line-search component of L-BFGS can lead to a performance increase in both accuracy and speed. There is also an opportunity for speedup in the quantum target function evaluation, through improving the performance of costly matrix functions.

Parallelisation will be achieved through the implementation of GPGPU programming techniques, where a graphics processing unit (GPU) is used to run algorithms over hundreds of lightweight threads simultaneously. This processing paradigm is best described by Flynn's Taxonomy: single instruction multiple data (SIMD), and is particularly useful when performing a single operation over large datasets such as matrices or vectors.

Aims and Objectives

The aim of this project is to produce a framework that can be called from MatLab, which enables the parallel implementation of L-BFGS and the quantum target function. This implementation will interact with MatLab using the MEX interface which will be used to call the C and CUDA functions. The performance increase will come from improving current solutions by parallelising existing techniques.

- Implement the L-BFGS algorithm for use with a quantum control problem.
- Implement a CPU-serial quantum control problem and target function, with the focus on spin networks.
- To improve the performance of the L-BFGS algorithm by parallelising the line-search on the GPU.
- To improve the target quantum evaluation function, by implementing costly matrix functions on the GPU.
- To measure and critically evaluate the performance gain from parallelising the algorithm, it is expected that at small problem sizes, the improvement will be marginal.

Assumptions

The core aim of this project is based around improving the performance when calculating the effect of a single control/operator on a quantum system. I have assumed that by implementing parallelisation techniques, this will lead to an improvement in performance. The major areas for performance improvement in this project are the line search component of the L-BFGS algorithm, and parallelising the calculation of matrix exponentials as part of the target function.

The project is based around the time-dependant Schrödinger equation, which describes system state given the control hamiltonian, system hamiltonian and the target state. The equations are pre-existing, and the focus here is to improve techniques when calculating the effect of control / operations on the system. L-BFGS is implemented in this project by optimising control amplitudes, this algorithm is considered one of the best in its class, given the problem.

- The L-BFGS algorithm can be improved by parallelising the line search in both accuracy and speed.
- The L-BFGS and quantum target function evaluation performance enhancements will be great enough to warrant parallelisation.

Constraints

The main constraint in this project is the time limit, as this field of research is extremely large. It will be important to stay focused on the main objectives as there are many possibilities that can be explored in terms of improvement in both accuracy and speed. However, due to the size of the field, it is also possible to change focus during the project if the need arises to achieve a similar outcome; such as finding other areas to improve performance.

Legal and Ethical considerations

As this project does not require or handle sensitive data relating to individuals, or involve human or animal test subjects there is no need to acquire ethical consent on this project. However, when using external libraries in the project, it is important to adhere to copyright or licence laws that apply to the library. It is also important to note that in the long term the capabilities of quantum computers may bring ethical issues, such as many encryption standards may become insecure in the future, allowing access to private and sensitive data.

Deliverables

At the end of this project, I expect to have produced the following:

- A working and accurate demonstration of the L-BFGS applied to a quantum control problem.
- A working and accurate demonstration of the L-BFGS applied to a quantum control problem where the line search and parts of the quantum target function are conducted in parallel.
- Program functions for both methods are callable from MatLab.
- An evaluation of performance between the parallel and sequential versions of the framework.

Work plan

It is assumed that supervisor meetings will happen once per-week, however as my supervisor is currently on research leave this may not always be possible. In the cases where a meeting is not possible, any communication will be over email for that week. I also intend to keep detailed notes at the end of each week, which will form the basis of the report.

Teaching Week	Target	Key Milestones
2	Initial research of L-BFGS algorithm and quantum target function for spin networks	
	Initial framework of final report, where sections are clearly defined	
3	Implement L-BFGS and test on basic problems	Start of serial implementation
4	Develop quantum target function	
5	Adapt L-BFGS for quantum control of spin networks	
6	Test algorithm with Quantum Target Function	End of serial development
	Benchmarking algorithm including collection of results	
	Completion of sequential implementation	
7	Research CUDA implementation techniques, and become familiar with CUDA programming	Start of parallel implementation
	Research into parallelising line search and target function	
8	Development of L-BFGS parallel line search	
9	Development of parallel target function	
Easter Week 1	Development of parallel target function	
	Test algorithm with Quantum Target Function	
Easter Week 2	Completion of parallel implementation	End of parallel implementation
Easter Week 3	Benchmarking algorithm against parallel version	
10	Report Writing	Focus switches to report writing
11	Report Writing	
12	Report submission	Report submission

Tools

SOFTWARE

This system will be designed to be implemented with MatLab, it was chosen due to the efficiency of the inbuilt functions which may be used during the project.

VERSION CONTROL

As this project will have an iterative development process, I felt it was necessary to use GIT to keep track of changes throughout the process. I have chosen to use GitHub as I have past experience with the interface. The use of a GIT will aid in report writing as the differential in versions can be used to show clear progress towards a goal, and how this was achieved.

CODE QUALITY

As this project may be used by others in the future, it is important to develop code that is human readable. To aid this, I will be following naming conventions and coding styles that are applicable to that language (C, CUDA).