Automatic n-Buffering for Big Data processing

Asbjørn Thegler

October 2015

Abstract

WRITE LAST
During the research, it was discovered that using three buffers was an arbitrary constraint, and concluded that many use cases would benefit from a different number of buffers.

Contents

1 Introduction

Big Data has become a huge topic over the last few years. This new discipline entails processing very large amounts of data which is not a trivial task, even for computer scientists. A method known as *Triple Buffering* is often used by

scientists, and the method is often implemented exclusively for every single project. This has resulted in countless implementations, which are created with limited reusability. To amend this problem, this project will supply a generic implementation for use in research and industry.

1.1 Motivation

1.1.1 IO problems

Why do we want to buffer, instead of doing the naive thing?

1.1.2 Use Cases

1.1.3 Triple- vs n-buffering

Refer to "Theoretical Speedup"

1.2 Background

1.2.1 Big Data and Complexity

Complexity of algorithms doesnt matter with small data-sizes. Big Data makes complexity really important.

2 Theory and Analysis

2.1 Concurrency

2.1.1 Flow and Deadlocks

2.2 Theoretical Speedup with threads

2.3 Theoretical Speedup with processes

3 Design and Implementation

3.1 Multithreading with POSIX

3.2 Multiprocessing with OpenCL

4 Experimentation and Benchmarking

5 Conclusion

6 Future Work

6.0.1 IO throttling

The End;

Throughout the project I will focus on gaining knowledge on the following topics:

- Programming for and with Big Data
- Thoroughly understand and design generic synchronization
- Reason about complexity in relation to Big Data
- Reason about IO problems
- Design correct benchmarking