A Survey of Parallel Programming Concepts

Kyle Clapper

September 12, 2023

1 Introduction

Brief explaination of what parallel programming is and motivate why we need it.

2 Parallel Programming Concepts

This section will discuss the various parallel programming concepts and paradigms.

2.1 Concurrency vs Parallelism

Short explaination of the difference.

2.2 Shared State vs Message Passing

Some languages rely on shared state need mechanisms like locks and semaphores to manage it.

Other languages use no shared state and rely on message passing to share data between parallel tasks.

2.3 Task Parallelism vs Data Parallelism

This is basically the difference between GPU parallelism and CPU parallelism. SIMD vs running different tasks on different CPUs.

3 Concurrency in Modern Languages

This section will be an overview of the various parallel programming concepts modern day languages use. Each section will have an example of a typical (toy) parallel program written in one of these languages. Each section will talk about what types of parallel programming tasks each language/system is and isn't good for.

3.1 C, C++, and Java

Shared state and traditional CPU parallelism problems.

3.2 CUDA, OpenGL, and OpenMPI

Modern day GPU programming

3.3 Erlang

Message passing and no shared state.

3.4 Rust and Go

Not sure, but I know both of these languages promise to make parallel computing effortless. I think they both are a combination of C and Erlang style parallelism.

3.5 Javascript

Not parallel but concurrent (with the exception of web workers). Talk about the event loop and maybe talk about web workers too.

3.6 Racket

Go over Racket's implementation of threads and the Concurrent ML concepts they employ.

4 Current State of the Art

Depending on time, this section would talk about some of the research underway on new parallel programming paradigms, techniques, and technologies. Maybe mention Sam Caldwell.