# Distributed Systems Job Scheduler Using “Perfect” Fit Algorithm

Luke Glover, 4595587

### Introduction

A distributed system is a computing system where multiple nodes are coordinated so as to appear as a single system to the end user. Where different servers have different capabilities in regard to CPU cores, memory and disk capacity, scheduling becomes a key problem. Scheduling algorithms can optimise for different variables, including server rental cost, turnaround time or resource utilisation.

Following the implementation of a job scheduler using an “all to largest” scheduling algorithm in stage 1, this project proposes and implements a new scheduling algorithm, named (perhaps presumptuously) perfect fit. Whilst this algorithm is not designed to make perfect scheduling decisions, it does aim to completely fill servers leaving no CPU cores idle on running servers.

### Problem Definition

### Algorithm Description

The so-called perfect fit algorithm seeks to minimise the number of cores that are active but not currently executing a job. This would occur, for example, if a two core and a one core job were scheduled on a four core machine, leaving one core unused.

The algorithm does this by performing two scheduling decisions. The first is performed similarly to the best fit algorithm, finding the smallest server capable of running the job. However, unlike the best fit algorithm, perfect fit does not consider any already running or already queued jobs.

If the new job can immediately run on the server, it is scheduled there. However, if the job cannot immediately run on the server because it does not have available resources, the algorithm combines the new job with the running jobs to create a “super job” and then performs a second best fit with this new super job.

For example, imagine a system with two single core servers (server A and server B) and a two core server (server C) as shown in figure 1 below. A job requiring one core is running on