# Scheduling in Distributed Systems
# COMP3100 - S1 2022

This project aims to develop a job scheduler for distributed systems, e.g., compute clusters and (cloud) data centres. The project is set to be **individual** work, and it consists of two stages.

- Commencement: Week 1
- Weighting: 40% (Stage1: 20% and Stage 2: 20%)
- Programming language to be used: Java
- System to be used: Ubuntu Linux
- Project management[1]: Git (GitHub/Bitbucket)
- Report writing[2]: $\LaTeX$
- Stage 1 due: **5pm** Monday, Week 7
- Stage 2 due: **5pm** Monday, Week 13

Distributed (computing) systems come in various sizes and scale. They range from a single workstation computer with several processors, a cluster of compute nodes (aka servers) to a federation of geographically distributed data centres with millions of servers (e.g., Google data centres, AWS clouds and etc.). The main component of distributed systems is servers. These servers are networked together and often form a single system, e.g., a compute cluster or data centre. Roughly, when these servers are virtualised, the system is called a 'cloud' data centre or simply a cloud. Note that servers of a particular distributed system cannot be assumed to be identical/homogeneous; they are often heterogeneous in terms of processor architecture and resource capacity.
A distributed system, in general, deals with a diverse set of jobs of multiple users. In other words, servers in a distributed system are shared among many jobs, ranging from as simple as Linux commands like ls and gcc to complex data processing applications like MapReduce and Monte Carlo simulation jobs to long-running web services. These jobs have different resource requirements in terms of the number of CPU cores, memory and disk space.
Scheduling is the key technique for ensuring the efficient use of computer systems including distributed systems. In this project, you will be eventually developing a new job scheduler in a simulated distributed system. The simulation is enabled by a discrete event simulator (***ds-sim***). In COMP3100/COMP6105, the simulation adopts the client-server model. The server-side simulator has already been developed and provided to you. It oversees the simulation. In particular, it simulates a distributed system with user specified configurations (e.g., ds-sample-config01.xml) and pairs up with the client-side simulator containing one or more scheduling policies/algorithms. The IP address and default port number to be used to connect the server-side simulator are 127.0.0.1 and 50000, respectively. More detailed information on ds-sim can be found in https://github.com/distsys-MQ/ds-sim.

## Stage 1

Your task in this stage is to design and implement a *vanilla* version of client-side simulator (your own ds-client) that includes basic scheduling functionalities and a simple job dispatcher.

## Stage 2

Your task in this stage is to design and implement one or more new scheduling algorithms that shall be assessed in comparison with a set of baseline algorithms including First-Fit (FF), Best-Fit (BF) and Worst-Fit (WF).

** The working of your ds-client in each stage will be assessed in a demo session during a designated workshop, in addition to more detailed offline assessment on your submission.*
** Detailed instructions of each of these stages including the marking rubric will be available in a separate document.*

[1]There will be project management marks to be awarded based primarily on the commit history in your project git repository. The commit history shall show good project management practices, e.g., constant/regular commits throughout the duration of project.
[2]A $\LaTeX$ template will be provided.