# CS454 AI Based Software Engineering

Autumn 2021

## Assignment #2: Stochastic Optimisation
## Due by 23:59, 4 October 2021

### 1 Aim

Choose a real world problem that can be formulated as a metaheuristic search, and implement a solver.

### 2 Qualification & Justification

This assignment is intentionally open-ended. Choose a real world problem that you think is too big to solve precisely, and implement a solver that is based on a metaheuristic algorithm. For example, any large scale combinatorial optimisation (knapsack, scheduling, set cover, etc - see `https://en.wikipedia.org/wiki/Combinatorial_optimization`) that you can find in real life would qualify. It can also be about finding a suitable game strategy, finding a suitable course registration & time table, or solving puzzles. For this assignment, the topic does not have to be about software engineering ( later, your project has to be about software engineering). If you are unsure about your chosen problem, discuss it with me.

You should try to analyse the computational complexity of the problem you chose, to justify the use of a metaheuristic algorithm. If a simple iterative algorithm scales well for the problem, there would be no need for a metaheuristic approach.

### 3 Deliverables

Each person should submit the following deliverables by the submission deadline:

- **Report**: include a written report that contains detailed descriptions of your problem, your justification for the use of metaheuristic, and how you formulated the problem as a search. Describe the optimisation you have implemented in as much detail as possible. There is no page limit.
- **Implementation**: solver for your problems, self-contained in separate directories (see below).

For ease of marking, follow the following directory structure, and submit a zip file containing the top level directory, through KLMS.

```
[your student number]
├── report.pdf ..... Your report documenting both implementations
└── solver/ ..... a directory containing the solver and dependencies
```

### 4 Guidelines

- **Obviously**, do your own work. We take academic integrity very seriously. Any wrong-doing will be penalised accordingly.

- You do have to implement the optimisation algorithm **yourself**; do not use pre-developed frameworks.
- Make sure your submission is self-contained. It should not depend on any file outside the submitted directory, such as files on your own hard drive or online. We expect the solvers simply to work out of the box. If you use Windows machines (or, in fact, any machine of your own), we strongly recommend that you test the submission on a separate machine, in order to test whether it is relaly self-contained.
- Do reasonable documentation, so that we can use your solver for grading.