# SET10117 Multi-Agent Tutorial Timetable Generation

Calum Hamilton

40205163

### Introduction:

In this coursework, I was assigned the task of prototyping a decentralised multi-agent system which would assign timetable slots to students based on their preferences. Timetable slots are initially assigned to student agents (acting on behalf of a student) who attempt to swap their slot until they receive preferential times.

### Design:

My program design had to address the following five points:

1. A meaningful and relevant ontology, allowing agents to communicate
2. A communication protocol, allowing agents to communicate without revealing their preferences
3. A utility function, allowing the student to know when they’re satisfied with their slots
4. A strategy to determine which exchange requests to make, which to accept and which to reject
5. A metric to evaluate system effectiveness in terms of satisfying student preferences

I examine these design points below.

#### Ontology Design:

I designed my Ontology while focussing on passing the minimum number of messages possible. I include my Ontology Design as Appendix A and a more in-depth description of the Ontology Elements in Appendix B.

### Evaluation:

An area where I could have explored and tested more operators was the crossover operator. I struggled with choosing an appropriate selection operator due to the chromosome design. Operators tested outside of one-point, two-point and uniform crossover needed to be custom written to deal with the range of accepted values for each gene, and the 11th gene wrap from the Defender to Striker range. I tried using several other standard Deap operators, but the performance was poor, or they simply didn’t work.

Additionally, I could have tested other Algorithm types than the Hill Climber. I know that Tabu search and Simulated Annealing tend to perform better than Hill Climbing, and likely these algorithms could have achieved better results. I felt however, that I didn’t have time to fully investigate these different Algorithms, on top of the operator selection and exploration for each algorithm. I would recommend this as future work.

One strength of the task was the chromosome design that I created for the task. The initial idea and design, based on research of similar projects, gave me strong initial results. It highly simplified the expansive search space and generated solid initial results with the simple Hill Climber algorithm.

The updated version then further restricted the search space and eliminated further invalid permutations, making the problem like that of a knapsack problem.

Another strength was the scientific approach that I took. Firstly, I ensured my experiments were all conducted fairly and equally. I also ran experiments them enough runs to ensure the statistical significance between operators could be proven.

I also ensured that after fully testing each operator, I only updated one operator at a time. For example, I completed testing and experimentation on the mutation operator and selected the best operator, before moving on to looking at selection operators. This ensured that individual increases in performance could be specifically attributed to the operator.

Finally, I feel that I drew the correct conclusion from the two tournament sizes available. I feel this was because I fully understood the problem specification and viewed the solution with respect to the original brief.

Overall, I feel I approached this task in a solid manner and feel that the report and approach is appropriate for the task at hand.

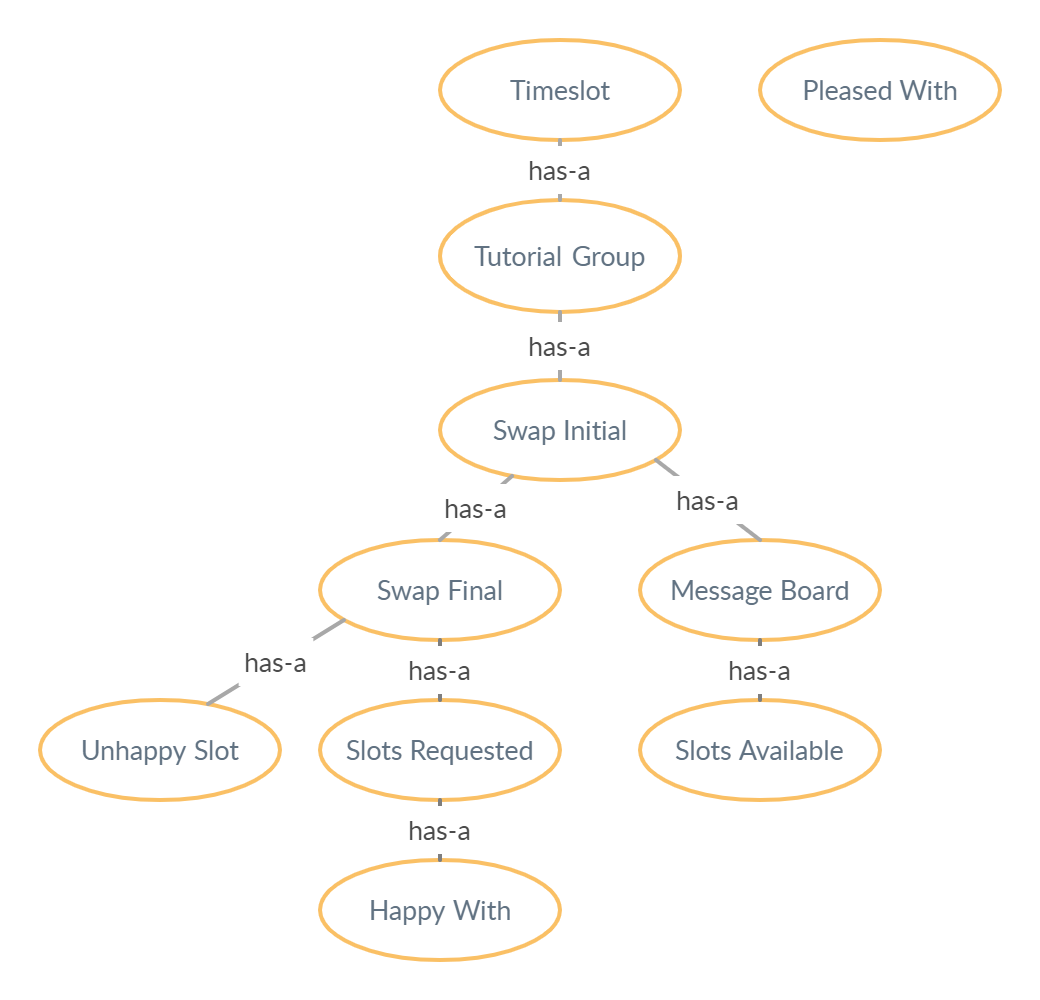
For future work I suggest testing other algorithm types than the Hill Climber and custom writing additional crossover operators for testing.

### References

1. S N, Omkar. (2003). Cricket team selection using genetic algorithm.
2. Ahmed, Faez & Deb, Kalyanmoy & Jindal, Abhilash. (2013). Multi-objective optimization and decision making approaches to cricket team selection. Applied Soft Computing. 13. 402–414. 10.1016/j.asoc.2012.07.031.

### Appendix:

#### Appendix A – Ontology Design



#### Appendix B – Ontology Elements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ontology Element** | **Properties** | **Data Type** | **Value Restrictions** | **Type** |
| Pleased With | Student Identifier | AID | Mandatory | Predicate |
| Timeslot | Day  Time | Integer  Integer | Mandatory  Mandatory | Concept |
| Tutorial Group | Timeslot  Tutorial ID  Class Size | Ontology Element  String  Integer | Mandatory  Mandatory | Concept |
| Swap Initial | Tutorial Group  Agent From | Ontology Element  AID | Mandatory  Mandatory | Agent Action |
| Swap Final | Swap Initial  Tutorial To  Agent To | Ontology Element  Ontology Element  AID | Mandatory  Mandatory  Mandatory | Agent Action |
| Unhappy Slot | Swap Initial | Ontology Element |  | Agent Action |
| Slots Requested | ArrayList<Swap Final> | List<Ontology Elements> | Mandatory | Predicate |
| Happy With | ArrayList<Swap Final> | List<Ontology Elements> | Mandatory | Agent Action |
| Message Board | ArrayList<Swap Initial> | List<Ontology Elements> |  | Concept |
| Slots Available | Message Board | Ontology Element | Mandatory | Predicate |