**Team 13 – Project Idea: University Course Scheduling**

***Problem description (from the Team Project Guidelines, underlining added by me)***

*This phase is the most important of the project, given that in this stage, you should be able to state what your system will do. Identifying the real-world problem that this system will give solution also needs to be described. The input and output of the system need to be listed.*

*The scope of the project should be well defined as well, where it is not too narrow or broad—as an example, building a system to solve the Deep Fake Detection Challenge (DFDC) that is sponsored by Facebook and Microsoft is too broad. Although, an implementation of the available tools/libraries already coded and provided, such as the Chatbot (Watson API) will be too narrow.*

*The measures to evaluate the success of your system also need to be defined in this phase. For this, you need to obtain a reasonable size dataset of example input-output pairs, either from existing sources or collecting one from scratch. The size of the dataset will depend on your specific problem.*

**General Problem Description**

The problem in general is to construct a schedule of classes at a university for a single term / semester. This schedule is often referred to as a “university timetable” and it includes four key elements: courses, lecturers, classrooms, and time slots. The courses, lecturers and classrooms have additional information (such as classroom capacity, course maximum enrollment, lecturer maximum teaching load, etc.) which form a set of constraints on the overall schedule.

This is generally considered a constraint satisfaction problem, with multiple hard constraints as well as some soft constraints or preferences. Although there appear to be some available open-source and commercial applications that will generate a timetable, scheduling in general is not a solved problem and is an active area of research in the AI community.

**Proposed Scope for Our Team Project (Goal)**

Depending on the size of the university, we could have a very large CSP to solve. For example, at Purdue University, in a typical term there are approximately 9000 classes offered in over 500 teaching spaces. The classes vary in size as do the classrooms. Some of the classroom spaces are shared across the different departments, while others are dedicated to a specific department.

For our project, I would suggest that we start with a more limited data set, something that maybe represents the course plan for just a College of Engineering. The goal would be to take a set of input data (a set of courses, lecturers, classrooms, and time slots) and produce a course schedule that satisfies the constraints.

To demonstrate the utility of our system, we would want to change some of the inputs and see the effect on the generated plan.

We would want to investigate a number of approaches to solving the problem, assess the run-time performance of each and compare the generated plans in some manner (to do that we would need to define a score or heuristic for how “good” a plan is).

It might be worth contacting the administration of the UTSA School of Engineering to determine how they develop their schedule of courses and see if there is any data available that we might be able to use.

Notes:

* A twist on this idea would be to look at sports scheduling; there are a lot of papers in that domain, but I am not sure about data sets
* Another twist would be to look at course scheduling from a higher level. Rather than worrying about a schedule during one term, we could look at when courses needed to be offered over a multi-year time scale in order to provide a reasonable path to degree completion. Again, I am not sure about available data sets.

**Potentially Applicable AI Technologies and Approaches**

* Constraint Satisfaction techniques, including all the tricks (backtracking search, minimum-remaining value, degree and least constraining variable heuristics, node/arc consistency, etc.).
* Other approaches to finding a solution include genetic algorithms and evolutionary/nature-inspired algorithms (such as ant colony optimization, bee colony, etc.). I think there are enough options to keep all of us busy.
* There are also some multi-agent approaches (where an agent represents a course, or a lecturer, or a classroom, and the multiple agents have to arbitrate and negotiate) which might be worth a look.
* In general, there are many possible solutions to a timetabling problem, so it might be interesting to come up with a mechanism to rank or score them, or perhaps even a heuristic that could be used in finding the N-best solutions.

**Data Sets / Sources**

The best website I have seen so far is UniTime.org. It includes excellent descriptions of the overall problem (as well as a couple of adjacent sub-problems). It provides a lot of benchmark data sets. It appears to provide access to a system that will produce a timetable.

Another good source is the International Timetabling Competition (2002-03, 2007, 2011, 2019). Datasets are available from the 2007 and 2011 competitions, and in general the datasets are smaller and more manageable that the ones at UniTime.org

* <http://www.cs.qub.ac.uk/itc2007/>
* <https://www.utwente.nl/en/eemcs/dmmp/hstt/>

**Schedule – TBD**

I would suggest that our plan of attack would be to acquire / define a small data set and then stand-up a number of algorithms / approaches to produce a timetable. Once that is done, as time permits, we could move toward larger data sets and examine how well the algorithms scale.