# Project Summary

*Short summary of the project setting.*

# Propositions

*List of the propositions used in the model, and their (English) interpretation.*

# Constraints

*List of constraint types used in the model and their (English) interpretation. You only need to provide one example for each constraint type: e.g., if you have constraints saying “cars have one colour assigned” in a car configuration setting, then you only need to show the constraints for a single car. Essentially, we want to see the pattern for all of the types of constraints, and not every constraint enumerated.*

# Model Exploration

*List all the ways that you have explored your model – not only the final version, but intermediate versions as well. See (C3) in the project description for ideas.*

A screen shot of a computer

Description automatically generated

Here, I noticed there were no TAs. It occurred to me that I need a constraint to say how many Tas we need for a particular course. This led me to the following constraint…

# Jape Proof Ideas

*List the ideas you have to build sequents & proofs that relate to your project.*

# Requested Feedback

*Provide 2-3 questions you’d like the TA’s and other students to comment on.*

# First-Order Extension

*Describe how you might extend your model to a predicate logic setting, including how both the propositions and constraints would be updated.* ***There is no need to implement this extension!***

## Domain of Discourse

* Natural numbers (used for levels)
* Objects for profs
* Objects for students (TAs)
* Objects for courses

## Predicates

* Assigned(x,y): Student x is assigned to course y for a TA’ship
* Preference(x,y,z): Student x has a preference level of z (1->5) for course y
* ProfPref(w,x,y,z): Prof w has a preference for TA x at level z (1->5) for course y
* MaxGrad(x,y): The maximum number of graduate students TA’ing course x is y
* Assume that we have equality for objects (e.g., or )
* …

We may want to specify the types of individual objects, so that the quantification is a little more oriented to the objects we use. These would be the types for this particular project:

* Student(x): x is a student
* Prof(x): x is a prof
* Course(x): x is a course
* Num(x): x is a number

## Functions

## Constraints

* All TAs can be assigned to only one course  
     
    
  If we want to make sure that the objects are of the correct type, then the formula would be:

Because it makes the formulae unwieldy to always include the types of the objects we quantify over, we will assume that they are implicitly included whenever we have a quantifier.

* No course gets a TA ranked 2 or lower by the instructor
* Some profs can veto certain TAs  
   Interpreted as: a prof can forgo assigning *any* level to a particular TA (across all courses)  
     
    
   If instead, we interpret it as having a value of 1 for the TA
* Nash equilibrium: no swap of TAs/Courses should lead to a better outcome (i.e., more preferences satisfied)

## Theorems

* If a prof didn’t rank a student 1-4, then they must have ranked them 5  
   …

# Useful Notation

*Feel free to copy/paste the symbols here and remove this section before submitting.*