Journal Report 17

02/17/20-03/02/20
Bryan Lu
Computer Systems Research Lab
Period 2, White

Daily Log

Monday, February 17

Spent another day studying annotation examples of other problems in the dataset and understanding what sort and how much labeling each sentence requires.

Tuesday, February 18

Started new file to process all of the olympiad problems I set aside as testcases earlier this year, and worked on editing the text so that the syntax parsing would work on the problems I wanted.

Thursday, February 20

Worked a little on polishing Prob Theory notes. Also got a set of problems to be succesfully parsed for syntax, and extracted the list of tokens for each sentence in the question for annotations.

Monday, February 23

Skip day.

Tuesday, February 24

Minor debugging and cleaning of the lists of tokens extracted from each sentence, and continued studying original test cases to see how to annotate various sentence structures.

Thursday, February 26

Fully finished reformatting first olympiad problem into annotations, and began work on others over the weekend.

Timeline

Date	Goal	Met
2/10	Run annotation data combined with problems through code and turn them into semantic trees, reformat my annotated problems.	Finished (with a questionable level of success?), still working on reformatting my annotated olympiad problems.
2/17	Finalize annotation data for the olympiad problems and ensure that they produce valid, connected semantic trees.	Not sure if this latter part is a concern, but I've decided on a test set I want to get the rest of the process to work on and began annotating.
2/24	Annotate about half of the olympiad problems used as test cases in the correct format.	Difficult to get started, but I've got enough to work with for the next week.
3/2	Start using the Naive Tag Model with olympiad problems, fix any issues that may arise, add more cases.	N/A
3/9	Continue training the model, look into any deeper issues, continue adding more cases as needed.	N/A

Final Goal

Grade	Requirements
A	With a well-trained Tag Model (at least 15-25 problems of experience), allow a user to enter an olympiad geometry problem and get Asymptote code in return. Well-written paper, clear presentation, organized Github with important elements commented for others to pick up work.
B	The above, but with issues, such as: conversion from literals to Asymptote code not working, model not trained well enough, hard-coding for a specific problem. Paper, presentation, or Github lacking in at least one-two aspects.
C	Attempts to train a Tag Model, unable to produce Asymptote code. Poorly-scrapped together paper and presentation, Github disorganized and unreadable.

Reflection

In order to get annotations for the olympiad problems I had saved from a couple of months ago, I looked more into the dataset of annotations and associated sentences that I found. This was sort of a slow process, because not all of the questions have associated annotations, and not all of the annotations were terribly useful for the complex sentence structure in olympiad problems. I did have to add some extra relations into the ontology, but this shouldn't be an issue. I am a little concerned about whether the model will be able to come up with these fairly complex relations, and whether or not the amount of annotations I'm giving is sufficient, but I'll find out when I try to make the model.

Here's an example of another annotation of a problem I did this weekend:

Question: In a triangle ABC, the length of AB >the length of AC. The foot of the altitude from A to BC is D. The intersection of angle bisector of ABC and AD is K. The foot of the altitude from B to CK is M. Let BM and AK intersect at point N. The line through N parallel to DM intersects AC at T. Prove that BM is the bisector of angle TBC.

Annotations (indexed by sentence):

1. IsTriangle@2(triangle@3), Ge@9(LengthOf@6(line@8), LengthOf@11(line@13))
2. Is@9(FootOf@1(point@6, line@8), point@10)
3. Is@9(IntersectionOf@1(ABisector@4(angle@6), line@8), point@10)
4. Is@9(FootOf@1(point@6, line@8), point@10)
5. Is@5(IntersectionOf@4(line@1, line@3), point@7)
6. Parallel@4(line@1, line@6), PointLiesOnLine@2(point@3, line@1), Is@9(IntersectionOf(line@1, line@8), point@10)
7. BisectsAngle@5(line@2, angle@8)