Journal Report 2
9/9/19-9/13/19
Rahul Mittal
Computer Systems Research Lab
Period 2, White

Daily Log

Monday September 16

Decide to replace all formulae with single letters to get rid of the problem with Stanford parser including the words/symbols in the formulae as part of the sentence, rewrite the identifyActions function to work with the constituency tree instead of the dependency graph.

Tuesday September 17

Added case for passive voice when a math noun is being acted on, begin testing of the identifyActions function starting with the 'Proof' sentence of my sample proof, modify outer structure of function to deal with all types of nodes (ROOT, S for sentence, FRAG for fragment), apparently the Node$parent function gives an UnsupportedOperationException which is really stupid, change function to have as arguments all ancestors of a node instead of just the node, now see that node$indexOf gives the index of a node as a descendant of another node rather than as a child, change to node.children().indexOf(other node), now it runs without any exceptions and found one of the actions (there are three), make minor change that I forgot to note down, MacBook dies, find charging cable for my MacBook, yay it works for that sentence, it also works for the 'Theorem' part which is just one whole premise, now going to Colorado proofs, looking at the first one and starting to make necessary modifications, added case for 'There is/are/etc.', added case for one math noun acting on another math noun, modified how the actions found are printed out.

Thursday September 19

Think about cases where a formula isn't an expression but rather an equation or inequality that would be equivalent to an actual clause in English rather than just a noun, for now substituting them with 'x is equal to y' for example, the 'Theorem' part of Colorado proof 1 works, the whole proof works yay!!! Now onto Colorado proof 2, finish the whole proof in five minutes, ah but I missed like the last three sentences, there's a sort of more complicated verb structure so I change from using isMathVerb and isMathNoun functions directly to using a isMathVerbActingOnMathNoun function which recursively calls itself when necessary, add a case for since (which implies a deduction in the main verb but also implies a presmise in the preceding clause), waste time by forgetting that I already accounted for passive voice, go back through the whole second proof to make sure I didn't break anything, realize that I previously overlooked a since that was earlier in the second proof, but now it works so that's good, the last sentence of proof 2 has an either, I need to deal with that, but now there's a lockdown drill and now 2nd period's over so that's for next time.

Timeline

Date	Goal	Met
September 9	Run Stanford parser on a couple of proofs	No, but I ran it on mine and looked at a couple of proofs
September 16	Run Stanford parser on more proofs, write a function trying to separate sentences into clauses with different actions	No, abandoning that idea completely
September 23	Finish prototype of function identifying actions and test it on more sample proofs to see what I didn't account for	Yes, almost done with making it work for Colorado proof 2
September 30	Revise function identifying actions and continue process to the point that 95% of random proofs on the Internet produce the correct output	
October 7	Modify function identifying actions to include information about statements made, meaning what verb used, what nouns acting/being acted on	

Reflection

This week went essentially like I was expecting (or really hoping) it to go, with me starting with a basic prototype of the function then going through the proofs to make corrections and additions to the function. I might have been hoping to finish more than almost 3 proofs, however since I was consistently making progress during the whole week I think it went pretty well.

Here's the current iteration of the 'Proof' part of my sample proof:

Proof: We can choose b urns which will contain a ball in c ways, and the balls can be matched up with the urns in d ways, so the total amount is g.

You can see that all of the formulae have been replaced with letters, which incredibly simplified the constituency tree for the sentence. You might notice that I skipped the letter 'a', because 'a' is actually a word. I also skipped 'e', maybe because it usually represents the mathematical constant. I also skipped 'f' because 'f' seemed to cause issues for some reason, though it may have just been an unrelated typo that made me think that.

Here's the action identification output for that sentence:

```
PREMISE: (VP (MD can) (VP (VB choose) (NP (NP (NN b) (NNS urns)) (SBAR (WHNP (WDT w
PREMISE: (VP (MD can) (VP (VB be) (VP (VBN matched) (PRT (RP up)) (PP (IN with) (NP
DEDUCTION: (VP (VBZ is) (NP (NN g.)))
```

The format of the action identification output here is that the action is is 'action: node' where the node is shown the way that Stanford CoreNLP shows it, which appears to be just the whole subtree of that node. Here, we can see, the output is as expected.

Just to show more of the function's output, here's the first Colorado proof (which can be found at):

Theorem 1. For all $n \geq k \geq 0$, $\binom{n}{k} = \binom{n}{n-k}$.

Proof. We will show that both sides of the equation count the number of ways to choose a subset of size k from a set of size n.

The left hand side of the equation counts this by definition.

Now we consider the right hand side. To choose a subset of size k, we can instead choose the nk elements to exclude from the subset. There are $\binom{n}{n-k}$ ways to do this. Therefore the right hand side also counts the desired quantity.

And here's the modified version with the output after every sentence:

Theorem: For all b, g is equal to h.

```
PREMISE: (VP (VBZ is) (ADJP (JJ equal) (S (VP (TO to) (VP (VB h.)))))))
```

Proof: We will show that both sides of the equation count the number of ways to choose a subset of size g from a set of size n.

```
No actions =(
```

The left hand side of the equation counts this by definition.

```
PREMISE: (VP (VBZ counts) (NP (DT this)) (PP (IN by) (NP (NN definition))))
```

Now we consider the right hand side.

```
No actions =(
```

To choose a subset of size g, we can instead choose the h elements to exclude from the subset.

```
PREMISE: (VP (MD can) (ADVP (RB instead)) (VP (VB choose) (S (NP (DT the) (NN h) (N
```

There are d ways to do this.

```
PREMISE: (VP (VBP are) (NP (NP (NN d) (NNS ways)) (SBAR (S (VP (TO to) (VP (VB do)
```

Therefore the right hand side also counts the desired quantity.

```
DEDUCTION: (VP (ADVP (RB also)) (VBZ counts) (NP (DT the) (JJ desired) (NN quantity
```

Lastly, here's a sentence from the later half of the second Colorado proof since it took up a lot of time on Thursday:

Since the subset does not include s, all of its k elements are chosen from the remaining $n - 1$ elements of S. Therefore there are $\binom{n-1}{k}$ such subsets.

And here are the modified version of it and the output for it:

Since the subset does not include s, all of its k elements are chosen from the remaining h elements of j.

```
PREMISE: (VP (VBZ does) (RB not) (VP (VB include) (NP (NNS s))))
DEDUCTION: (VP (VBP are) (VP (VBN chosen) (PP (IN from) (NP (NP (DT the) (VBG remai
```

There are a couple of important thoughts I had:

After I made the various changes to acommodate that sentence in proof 2, I went back through previous sentences of proof 2 to make sure I hadn't broken anything. I hadn't, but it still made me realize that I could've and it might be useful to set up a sort of automated testing thing that I could run my code through after I make modifications so that I don't break things in the process of fixing other things.

My next thought was that, right now the output is literally just one of four values (the action) for each node that is identified as having an action, but given the way that I'm writing the function to do this, it wouldn't be much more difficult to be more specific about what actions are being performed, and that's something that will probably be useful or even necessary in the later stages of my project. Because of that I've dedicated the week after this coming week to modifying my function to give those additional details.

One last thing I was thinking about was that right now I literally just have hardcoded lists of math nouns, math verbs, math adjectives, etc. as they come up in proofs because the specific words aren't something I want to worry about right now, but eventually it'll probably be good to compile them in a more organized way, especially given that I'm going to need to ultimately translate their actual meaning. However, that's not something that poses an immediate problem, so I'm not going to worry too much about it for right now.