Journal Report 16
1/20/20-1/31/20
Rahul Mittal
Computer Systems Research Lab
Period 2, White

Daily Log

Tuesday January 21

Statement tagging function hanging on Colorado 1, remove request for outfile name (nothing being output to file right now), add initial debug of arguments, hanging before function is called, add debug of output in main, hanging before that even, maybe something with the tokenization changes?, the next line not being printed has comma, forgot j++, added it, now getting NoSuchElementException because of comma stuff, move PLACEHOLDERS string constant to BetterParser and use it there for the if statements to concisely make sure that the placeholder is a letter, rerun, forgot to add it for variable math objects, added and rerun, now getting IndexOutOfBoundsException in the tagging function, changed statementTags.size $<$ ix to statementTags.size $\leq$ ix, actually finished without exceptions, only has SentenceTags though.

Thursday January 23

Debugging k + arguments in dfs, forgot that everything was wrapped in a ParsedStatement, now automatically unwrap ParsedStatements while making sure that the ParsedStatement is used when adding to the statements list, appropriate ByTag and ThereforeTag appear now, in findSimilarSubstrings replace Pair<Int, Int> with SubstringIdentifier data class to clarify meaning of fields, look at Colorado 1 to see where implications come from, need to be able to determine what "desired quantity" etc. refer to, starting with desired quantity, similar flattening for intents would be convenient to have, added intents argument to getStatementsAndImplications function, created new section at end with desiredQuantity Quantity? variable, doing fun casework on each intent, for Determine just immediately declare it as the desired quantity, look up how loop labels work in Kotlin, decide to add check that Determine isn't determining a DesiredQuantity, realize that it will relieve a lot of unwrapping of Parseds if I simultaneously create a MathThing without any Parsed wrappers for each Parsed, doing that now, add parallel field for each Parsed class, make easier necessary changes in parseStatementOrIntent, add parallelParameters field to ParsePosition, add parallel: Parsed... -> ... function for easy unwrapping, make remaining changes, done with parallel stuff.

Wednesday January 29

Use the parallel intent in the intent loop in getStatementsAndImplications, add check that quantity of Determine isn't HangingAmount, decide that the ChooseFor intent doesn't have anything to do with desired quantity, change name of desiredQuantity variable to hangingAmount for consistency.

Thursday January 30

AMC 12A Review

Sunday February 2

Start Show case, use DFS again for flattening, copy over dfs function from the statements stuff, add Units at end of when cases because the Kotlin compiler is annoying, get rid of 90% of the code in the copied dfs function, figure out (not really, just see which one compiles) how in/out work in generics, make List of quantity classes that can be considered concrete combinatorial things as opposed to hanging things or formulas, currently containing Amount and Size classes, change check in Determine case from !is HangingAmount to is one of the concrete classes, in Show case, for each statement of flattened, if is instance of Equals, try to make q1 and q2 the desired quantity (if they're of a concrete class).

Timeline

Date	Goal	Met
January 13	Forgot to set a goal	Added some MetaStatement subclass + templates, started conversion of sentences to proper statements and implication finding
January 20	Finish conversion of sentences to proper statements, finish adjacent implications from "Therefore", find similar phrases used in different statements	Yes, yes, yes but haven't used them for anything yet
February 3	Be able to find all implications between statements in Colorado proofs	No, worked on figuring out hanging quantity
February 17	Be able to find all implications between statements in Colorado proofs, start work on combinatorics foundation in Coq	
February 24	Continue work on combinatorics foundation in Coq	

Reflection

I think the work I got done this week was pretty good, though none of it was what I had anticipated working on previously. I had realized while making my last journal that my code to flatten the statements and tag them was hanging, so I needed to fix that first of all, which was a fairly straightforward process though it did take a bit of time. Now that it's working, you can see the output for Colorado proof 1 below:

Modified version of Colorado 1:

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 $n-k$ elements to exclude from the subset. There are $n >> (n-k)$ ways to do this. Therefore the right hand side also counts the desired quantity.

And the flattened statements + tagging:

```
j = 0
statement = ParsedStatement(statement=Equals(q1=ParsedQuantity(quantity=SideOfEquat
tags = [SentenceTag(ix=0),
    ByTag(id=41b271cb-ee83-4d33-a65a-4d51dbf98905, concept=DEFINITION)]
j = 1
statement = ParsedStatement(statement=Equals(q1=Amount(obj=Choice(noun=ParsedMathOb
tags = [SentenceTag(ix=1)]
j = 2
statement = ParsedStatement(statement=Equals(q1=Amount(obj=ParsedMathObject(mathObj
tags = [SentenceTag(ix=2)]
j = 3
statement = ParsedStatement(statement=Equals(q1=ParsedQuantity(quantity=SideOfEquat
tags = [SentenceTag(ix=3),
    ThereforeTag(id=d0c84c9a-aa13-4fd1-8a46-5276d7c8db33)]
```

I didn't bother expanding out the statements this time because they're exactly the same as what I've probably included in past journals and would really just waste paper. Since none of the sentences in this proof contained compound statements like And, you can't really see the flattening here except that statements 0 and 3 have been unwrapped from their MetaStatements. The important thing here then is the tagging, which in addition to the SentenceTags shows how the MetaStatements containing statements 0 and 3 were converted to tags.

My new work this week was on determining the meaning of the HangingQuantity in a particular proof, which I realized that I needed to do after looking at Colorado proof 1. My method of doing this relies on the assumption that all HangingQuantities appear explicitly in an intent, so I just did casework on the different types of Intents (since I only have three of them) to determine if they contain a Quantity suitable as the HangingQuantity. I further made the assumption that if any of the top level Quantities in an Intent are suitable to be the HangingQuantity, then they are, which is I think a more reasonable assumption than it initially seems - proofs will generally avoid ambiguities that would contradict that assumption.

With those assumptions, my work boiled down to two things: First, I needed to determine what kind of Quantities are suitable to be the HangingQuantity. I decided that "desired quantity" and similar statements generally referred to a combinatorially defined quantity, rather than some kind of formula (and obviously not to another hanging-type thing). From my Quantity subclasses, only Size and Amount seemed to fit the bill, so I made a CONCRETE_QUANTITY_CLASSES list containing those two classes, and checked Quantities' classes against that list to determine whether they were suitable.

Second, I needed to actually extract the Quantities from the Intents to check them. This was trivial for Determine, or at least I thought it would be until I realized that since I have literally everything wrapped in some kind of Parsed I would need to do all kinds of unwrapping which I didn't want to do, so I took some time to make it so that each Parsed, in addition to containing a thing which would have more Parsed wrappers inside, would also contain a "parallel" version without any Parsed wrappers contained inside. Later parallels can be constructed using earlier parallels in the same way that later normal values are constructed using earlier normal values, so the implementation was just a reflection of my existing code for those values.

With that, Determine became legitimate trivial. I decided that ChooseFor couldn't yield something suitable for HangingQuantity, since there aren't explicitly stated Quantities, and for Show,

it seemed like Equals was the only Statement that really had Quantities at the top level, so I tested both sides of the equality in that case, finishing my determination of HangingQuantity.

It's good that all of that is done, but I now really need to finish implication finding this 2-week period, and hopefully also start the next phase of my project, considering how little time is left to start and finish something completely new. I think it's been a consistent theme this year of me realizing how much less time I have because I never account for shorter weeks like we just had something like three times in a row, but hopefully I can get acquainted with Coq quickly enough and do my work with it at a fast enough pace to finish in time.