**If you see something sus please please please start a discussion so we can get as close to the real solution as possible**

1.a)

Note: we include because from the negation of the positive example and the contrapositive of the rule we also entail that must also be modeled through .

Given the mode declaration, we must have at most: 1 q in the head, 1 r in the body and 1 s in the body.

So and

Consistency: doesn’t cover any of the negative examples

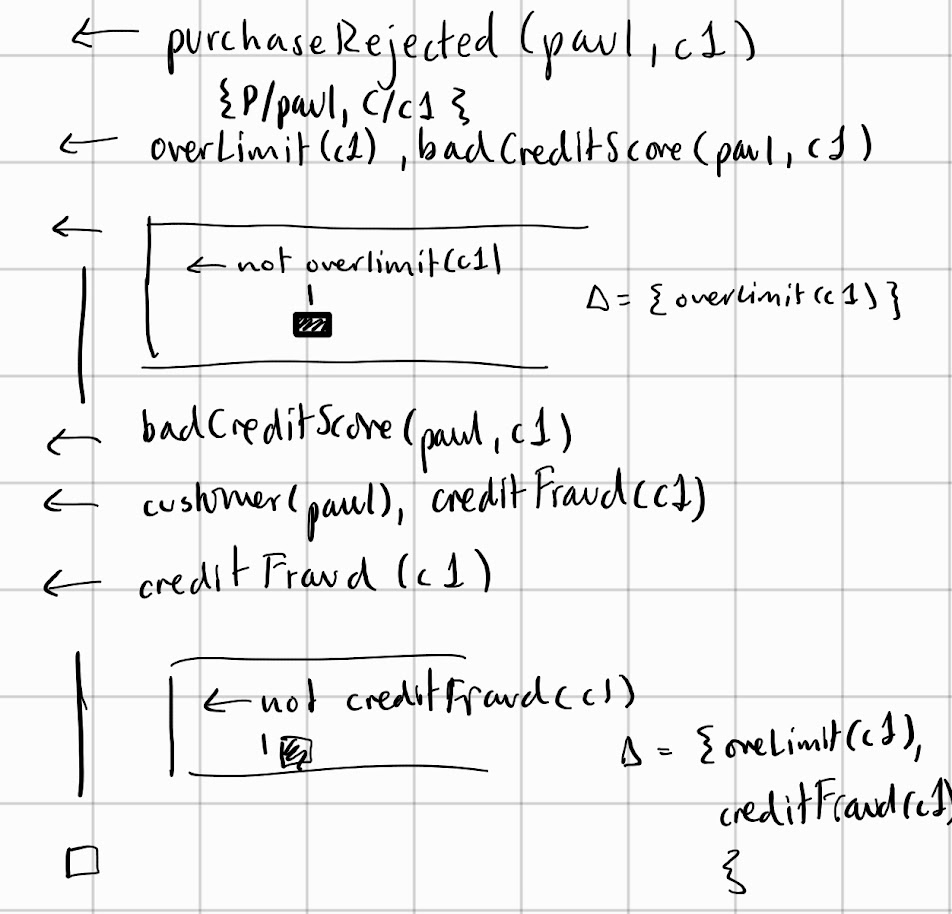
For there to be a H1 that is not consistent and subsumes Bot(B,p(a)) it needs to cover p(b).

~~Idea: but modeb says r has to be an output variable...~~

1.b.i)

Take a positive seed example as

Let with and

1. Abductive phase: we generate   
   
2. Deductive phase:  
      
   and K is the immediate generalisation. The order of these rules is important because owns(-,+) mode
3. Inductive Phase:  
   Given a goal H we can just see if H theta subsumes K.  
   Then   
   Therefore, H is a valid inductive solution for the task T.

1.b.ii)

In order to prove purchaseRejected we need overLimit and badCreditScore. In order to prove badCreditScore we need customer and creditFraud. The only things separating paul from jane and peter is that he has both reached his limit on card c1 and had his details stolen. So we should find him guilty of creitFraud if both of these are true. Only way to show that detailsStolen is to pair it with owns atom, so these two have to stay together or else we cant get c1.

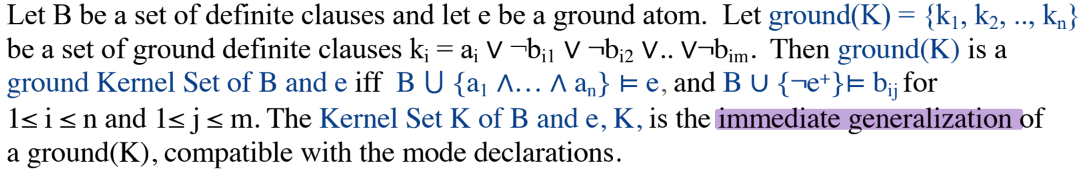
I see nothing wrong with just swapping the two head atoms so the body changes to its conterpart, since we need both overLimit and badCreditScore to reject a purchase, so it shouldn’t matter in which order we prove the two.

1.b.iii)

Progol5 is incomplete in terms of non-observational learning and we have learnt two clauses that don’t appear in the positive examples. Therefore, progol5 wouldn’t be able to learn the above as it would instead learn something with purchaseRejected in the head of the learnt rule.

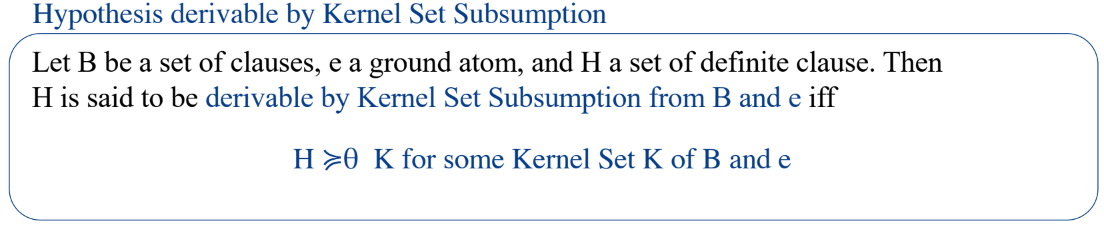
2.a.i)

From slides:



Therefore the ground Kernel will be of the form

Therefore, any clause that is derivable by Kernel set subsumption would be a clause H that theta subsumes Kernel(B, e^+):



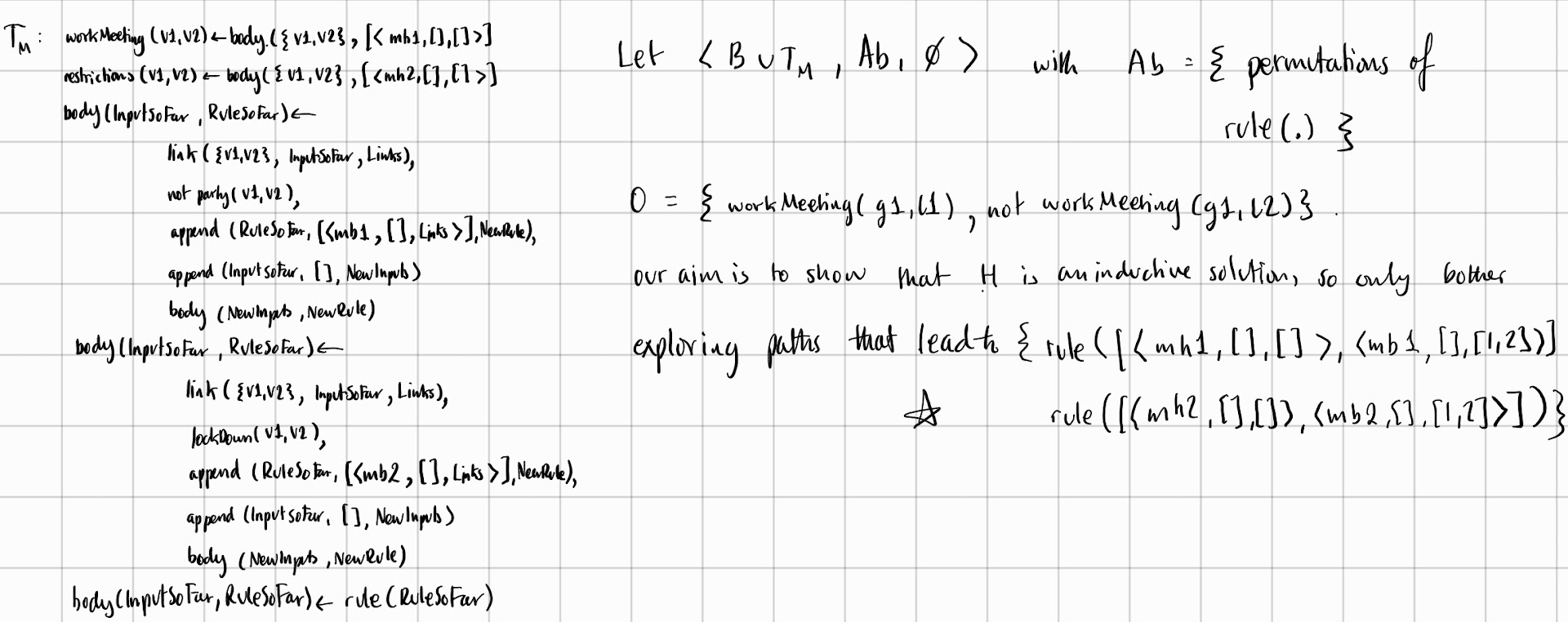
However, we won’t accept it because we cannot prove any of the positive examples from it; in order to prove r we need s or/and t. In order to prove t we need s. We don’t have a rule with s in the head or any s facts, therefore we cannot prove that

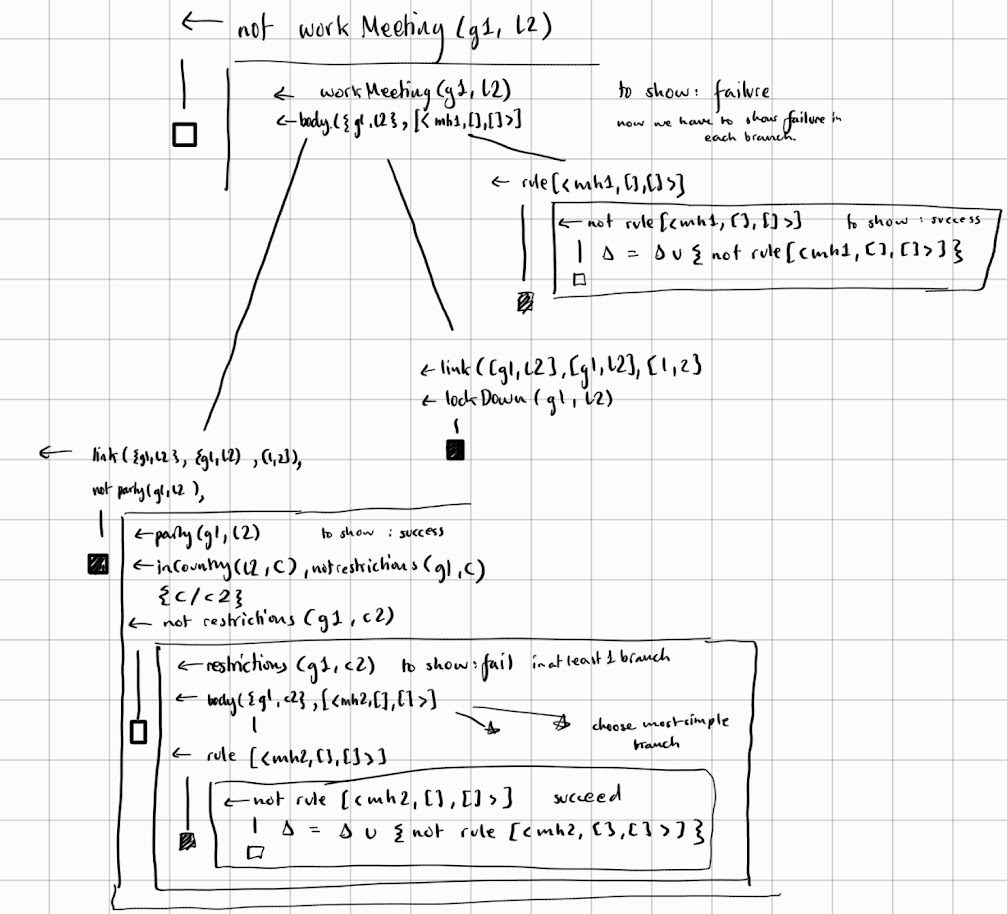
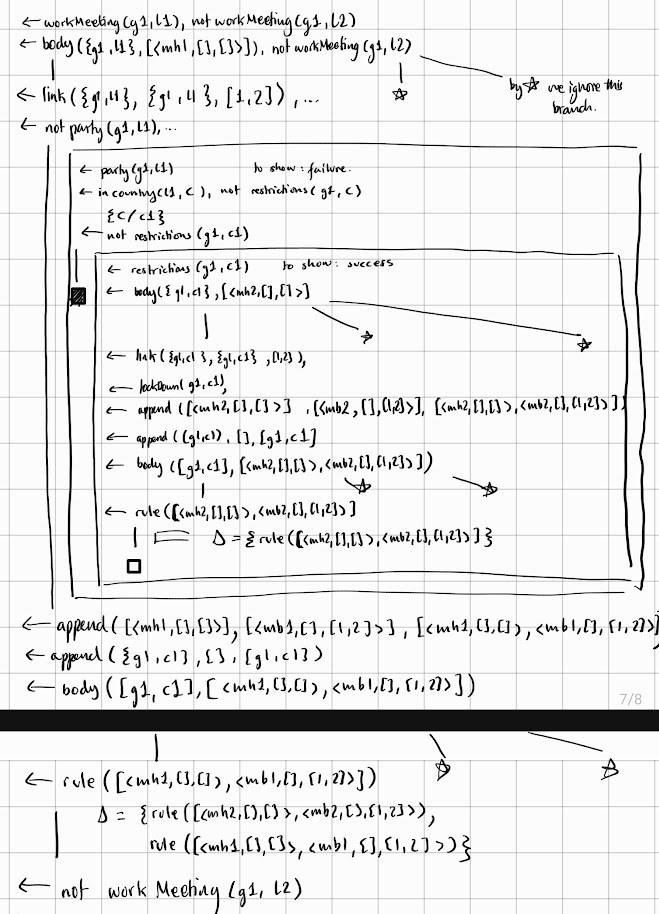
In the case where we straight up have r as a fact, from the abductive step of calculating the ground kernel set we get . The kernel set is therefore the immediate generalization r(X,Y) which we can use to prove the negative examples, so that’s no good either.

2a.ii)

Any ideas?

2.b.i)





Therefore the final set of abducibles are:

The first two of which, when translated back into our language are:

Which with the rudimentary substitution of V1/G, V2/L give the above H

2.b.ii)

Complete: entails all positive examples

Not consistnet doesn’t not prove all negative examples.

Just remove the body of the workMeeting rule?

3.a.i)

{{0,0}, {0,1}, {1,0}, {1,1}} so 4 worlds (these only depend on the facts)

3.a.ii)

Ony worlds where this is true is when we have {1,0}, {0,1}, {1,1}

3.b.i)

Answer: 1  
t2(a) t2(b) t1(1) t1(2) p(2) s(1,b) s(1,a) r(1,b) r(1,a) q(1,b) q(1,a)  
Answer: 2  
t2(a) t2(b) t1(1) t1(2) p(2) s(1,b) s(1,a) r(1,b) r(1,a) p(1)

3.b.ii) skeleton rule formation, so

B and

Q(v1,v2) :- t1(v1), t2(v2), rule(1)

Q(v1,v2) :- t1(v1), t2(v2), not p(v1) rule(2)

Q(v1,v2) :- t1(v1), t2(v2), r(v1,v2), rule(3)

Q(v1,v2) :- t1(v1), t2(v2), r(v1,v2), not p(v1), rule(4)

And

{rule(1), rule(2), rule(3), rule(4)}

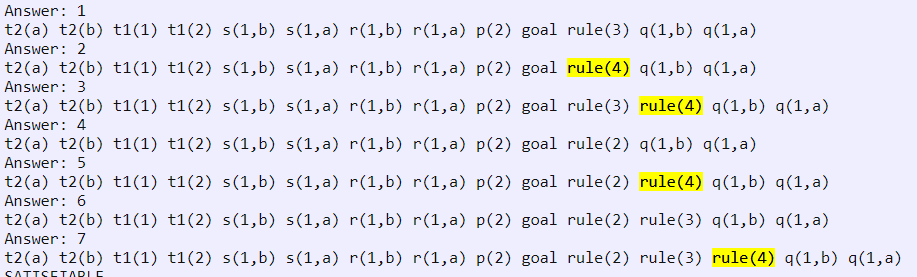
And

Goal :- p(2), not p(1)

:- not goal

3.b.iii)

Since we want to prove H (which I have labelled as rule(4)) we start off with rule(4) being in our answer set, including all of the facts in the background.

I did this by taking the relevant grounding and slowly working my way up to a stable model

3.b.iv)

No we can make rule(2) or rule(3) which in my encoding are rules with less literals in the body, I.e. shorter

3.b.v)

It is not, in pt (I) we found an answer set of B U H containing p(1), hence there’s an answer set which covers a negative example, hence not a cautious inductive solution.

From above we can see that its not, but why?