Midterm 2

15-317: Constructive Logic

November 5, 2009

Name: **Solution** Andrew ID: `wlovas`

Instructions

- This exam is closed-book, but one two-sided sheet of notes is permitted.
- There are five problems, each with several parts. Not all problems are the same size or difficulty. You have 80 minutes to complete the exam.
- When writing proofs, remember to label each inference with the rule used.
- You may find it helpful to construct your proofs on scratch paper (such as the back of a page) before writing it clearly in the space provided.
- Most important,

Don't Panic

Good luck!

	Problem 1	Problem 2	Problem 3	Problem 4	Problem 5	**Total**
Score						
Max	25	35	55	20	15	150
Grader						

1 Logic Programming (25 points)

Consider a predicate `intersect(+list, +list)` which holds of any two lists that share an element, defined using the `member` predicate on lists.

```
member(X, [X|Xs]).                 intersect(L1, L2) :-
member(X, [Y|Ys]) :-                   member(X, L1),
    member(X, Ys).                     member(X, L2).
```

Task 1 (10 pts). What mode or modes must `member` satisfy in order for `intersect` to satisfy mode `(+, +)`? Explain in detail why `member` satisfies this mode or modes.

Solution. `member` must satisfy two modes, `(-, +)` and `(+, +)`. The first subgoal treats `L1` as an input and `X` as an output, enumerating all members of `L1` through backtracking; the second subgoal treats both `X` and `L2` as inputs, checking that `X` is a member of `L2`.

We can show that `member` satisfies both modes by examining its clauses.

- Mode `(-, +)`:
  1. First clause: Assume `[X|Xs]` is ground. We must show that `X` is ground, and it is by assumption.
  2. Second clause: Assume `[Y|Ys]` is ground. We must show that `X` is ground. For the subgoal to be well-moded, we must show that `Ys` is ground, which it is by assumption. If the subgoal succeeds, we learn that `X` is ground, which is what we needed to show.
- Mode `(+, +)`:
  1. First clause: Assume `X` and `[X|Xs]` are ground. There are no output arguments and no subgoals, so there is nothing to show.
  2. Second clause: Assume `X` and `[Y|Ys]` are ground. For the subgoal to be well-moded, we must show that `X` and `Ys` are ground, which they are by assumption. ■

Task 2 (5 pts). Are there any other modes that `intersect` can be given, besides `(+, +)`? Explain your answer.

Solution. No, there aren't. A cursory examination of the clauses defining `member` shows that its second argument can never be an output—in the first clause, there is no way to know that `Xs` is ground, and in the second, there is no way to know that `Y` is ground. For intersect to output either `L1` or `L2`, the `member` predicate would have to output its second argument, so since it cannot, the only mode `intersect` can satisfy is `(+, +)`. ■

(Problem continues on next page)

Consider the following ways of introducing cuts into the code for `intersect` to cut off backtracking. We are only interested in the behavior on the given mode, where `L1` and `L2` are both inputs and valid lists.

Task 3 (5 pts). Is the following cut red or green? If it is red, exhibit a query provable *without* the cut that isn't provable *with* the cut. If it is green, explain why no such query exists.

```
intersect(L1, L2) :-
    member(X, L1),
    !,
    member(X, L2).
```

Solution. This cut is *red*: the cut will commit proof search to the first member of `L1` it finds, failing if the two lists intersect on any other member of `L1`. For example, the query `intersect([1,2], [2])` will fail, but it would succeed without the cut. ■

Task 4 (5 pts). Is the following cut red or green? If it is red, exhibit a query provable *without* the cut that isn't provable *with* the cut. If it is green, explain why no such query exists.

```
intersect(L1, L2) :-
    member(X, L1),
    member(X, L2),
    !.
```

Solution. This cut is *green*: since `intersect` has mode `(+, +)`, the only possible successful answer is "true", and the cut only commits once we have found a successful answer. Any further search could only discover another way the lists intersect, but we already know they intersect once we reach the cut. ■

2 G4ip (35 points)

Recall Dyckhoff's contraction-free sequent calculus **G4ip**.

Task 1 (10 pts). Give a **G4ip** derivation of the sequent $\cdot \longrightarrow \neg\neg(P \vee \neg P)$. Assume the proposition P is atomic and $\neg A$ is a notational definition, as usual.

Solution. Expanding all the negations into their notational definitions, we must prove $\cdot \longrightarrow ((P \vee (P \supset \bot)) \supset \bot) \supset \bot$. The proof follows, with the active formulae in each conclusion highlighted.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\bot, P, \bot \supset \bot \longrightarrow \bot}\ \bot L}{P \supset \bot, P, \bot \supset \bot \longrightarrow \bot}\ P{\supset}L \qquad \dfrac{}{P \supset \bot, \bot \longrightarrow \bot}\ \bot L}{P \supset \bot, (P \supset \bot) \supset \bot \longrightarrow \bot}\ {\supset}{\supset}L}{(P \vee (P \supset \bot)) \supset \bot \longrightarrow \bot}\ {\vee}{\supset}L}{\cdot \longrightarrow ((P \vee (P \supset \bot)) \supset \bot) \supset \bot}\ {\supset}R$$

■

(Problem continues on next page)

Consider a variant of **G4ip** with a "memoizing" right rule for conjunction.

$$\dfrac{\Gamma \longrightarrow A \quad \Gamma, A \longrightarrow B}{\Gamma \longrightarrow A \wedge B}\,\wedge R'$$

It can be difficult to tell whether such a rule represents an optimization or a complication of the decision procedure embodied by **G4ip**.

Task 2 (10 pts). Find a derivation of a sequent that "saves time" using $\wedge R'$. The smaller derivation should be identical to the larger aside from some work that is saved thanks to the memoizing rule.

Solution. There are many examples. Anytime more than one rule is required to prove an atomic proposition, memoizing can replace a proof of it with an immediate application of the init rule. For instance:

$$\dfrac{\dfrac{\dfrac{}{P, Q \longrightarrow Q}\,\text{init}}{P, P \supset Q \longrightarrow Q}\,P{\supset}L \qquad \dfrac{}{P, P \supset Q, Q \longrightarrow Q}\,\text{init}}{P, P \supset Q \longrightarrow Q \wedge Q}\,\wedge R'$$

■

The right subproof could have been the same as the left, but instead it was one shorter by immediately appealing to "init".

(Problem continues on next page)

Task 3 (15 pts). Find a derivation of a sequent that “wastes time” using $\wedge R'$. The larger derivation should be identical to the smaller aside from some extra work that could be done thanks to the memoizing rule.

Solution. There are again many examples. Almost any non-atomic formula “memoized” into the context can be needlessly broken down. For instance:

$$\dfrac{\dfrac{\dfrac{}{P \longrightarrow P}\,\text{init} \qquad \dfrac{}{P \longrightarrow P}\,\text{init}}{P \longrightarrow P \wedge P}\,\wedge R \qquad \dfrac{\dfrac{}{P, P, P \longrightarrow P}\,\text{init}}{P, P \wedge P \longrightarrow P}\,\wedge L}{P \longrightarrow (P \wedge P) \wedge P}\,\wedge R'$$

The right subproof could have concluded with an immediate application of “init”, but instead it broke down the memoized assumption first.

Although the extra work is never *necessary* to a proof—a short proof can always be weakened with an extra assumption—a given proof search strategy will not necessarily find the shortest proof. In this case, it is easy to imagine a theorem prover that would break down the new assumption just to be safe, since conjunction is asynchronous on the left. ■

3 Classical Sequent Calculus (55 points)

We may define a classical sequent calculus by allowing a sequent to have multiple conclusions. The main judgement now becomes $\Gamma \;\#\; \Delta$, with the interpretation that if all of the propositions in Γ are true, then at least one of the propositions in Δ is true, or equivalently: if all of the propositions in Γ are true and all of the propositions in Δ are false, then they are in contradiction.

$$\frac{}{\Gamma, P \;\#\; P, \Delta}\;\text{init}$$

$$\frac{\Gamma \;\#\; A, B, \Delta}{\Gamma \;\#\; A \vee B, \Delta}\;\vee R \qquad \frac{\Gamma, A \;\#\; \Delta \quad \Gamma, B \;\#\; \Delta}{\Gamma, A \vee B \;\#\; \Delta}\;\vee L$$

$$\frac{\Gamma, A \;\#\; \Delta}{\Gamma \;\#\; \neg A, \Delta}\;\neg R \qquad \frac{\Gamma \;\#\; A, \Delta}{\Gamma, \neg A \;\#\; \Delta}\;\neg L$$

We do not need rules for $A \wedge B$, $A \supset B$, $\top$, or $\bot$ because these can be defined in classical logic.

Task 1 (5 pts). Give a derivation of the sequent $\cdot \;\#\; P \vee \neg P$ in this calculus. Assume the proposition P is atomic.

Solution.

$$\cfrac{\cfrac{\cfrac{}{P \;\#\; P}\;\text{init}}{\cdot \;\#\; P, \neg P}\;\neg R}{\cdot \;\#\; P \vee \neg P}\;\vee R$$

■

(Problem continues on next page)

The Cut Principle for this calculus is as follows:

Theorem (Cut). If $\Gamma \mathrel{\#} A, \Delta$ and $\Gamma, A \mathrel{\#} \Delta$, then $\Gamma \mathrel{\#} \Delta$.

Task 2 (20 pts). Prove the principal case of the Cut Principle for $A \vee B$. Given

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1\\ \Gamma \mathrel{\#} A, B, \Delta\end{array}}{\Gamma \mathrel{\#} A \vee B, \Delta}\ \vee R \qquad \text{and} \qquad \mathcal{E} = \dfrac{\begin{array}{cc}\mathcal{E}_1 & \mathcal{E}_2\\ \Gamma, A \mathrel{\#} \Delta & \Gamma, B \mathrel{\#} \Delta\end{array}}{\Gamma, A \vee B \mathrel{\#} \Delta}\ \vee L,$$

come up with a derivation of $\Gamma \mathrel{\#} \Delta$. You may assume the following lemmas:

Lemma (Left Weakening). If $\Gamma \mathrel{\#} \Delta$, then $\Gamma, A \mathrel{\#} \Delta$ with a structurally identical deduction.

Lemma (Right Weakening). If $\Gamma \mathrel{\#} \Delta$, then $\Gamma \mathrel{\#} A, \Delta$ with a structurally identical deduction.

Solution. The desired result follows from Right Weakening and two appeals to the inductive hypothesis.

$\Gamma, A \mathrel{\#} B, \Delta$	By Right Weakening on $\mathcal{E}_1$.
$\Gamma \mathrel{\#} B, \Delta$	By i.h. on A, $\mathcal{D}_1$, and above.
$\Gamma \mathrel{\#} \Delta$	By i.h. on B, above, and $\mathcal{E}_2$.

■

(Problem continues on next page)

Recall that an *invertible* rule is one whose conclusion entails its premises. For the next two Tasks, you may assume the following structural properties of the classical sequent calculus as lemmas:

Lemma (Left Weakening). If $\Gamma \# \Delta$, then $\Gamma, A \# \Delta$.

Lemma (Right Weakening). If $\Gamma \# \Delta$, then $\Gamma \# A, \Delta$.

Lemma (Identity). For any Γ, Δ, and A, we can derive $\Gamma, A \# A, \Delta$.

Lemma (Cut). If $\Gamma \# A, \Delta$ and $\Gamma, A \# \Delta$, then $\Gamma \# \Delta$.

It turns out that all rules in the calculus are invertible.

Task 3 (15 pts). Prove that the $\vee R$ rule is invertible.

Solution. Suppose the conclusion holds: $\Gamma \# A \vee B, \Delta$. We must derive the premise: $\Gamma \# A, B, \Delta$.

First, by two applications of Right Weakening on the assumption: $\Gamma \# A \vee B, A, B, \Delta$. Call this derivation $\mathcal{D}$.

Then, we can derive the following using $\vee L$ and Identity:

$$\dfrac{\overset{\textbf{\textit{(Identity on } A\textbf{\textit{)}}}}{\Gamma, A \# A, B, \Delta} \qquad \overset{\textbf{\textit{(Identity on } B\textbf{\textit{)}}}}{\Gamma, B \# A, B, \Delta}}{\Gamma, A \vee B \# A, B, \Delta}\ \vee L$$

Call this derivation $\mathcal{E}$.

Finally, by Cut on $A \vee B$ and the derivations $\mathcal{D}$ and $\mathcal{E}$, we conclude $\Gamma \# A, B, \Delta$, as required. ■

(Problem continues on next page)

Task 4 (5 pts). We can interpret the inference rules as a proof search calculus, working bottom-up. Does proof search in this calculus always terminate? If so, explain why. If not, give a sequent that causes proof search to loop.

Solution. Yes, proof search always terminates: each rule's premises contain one fewer connective than its conclusion. Since each rule application eliminates one connective and sequents are finite, we always eventually reach a state where no more rules may be applied. ■

Task 5 (10 pts). Since we are in classical logic, we can define $A \supset B$ as $\neg A \vee B$. Give *derived* left and right rules for $A \supset B$ based on this definition.

Solution. By applying the rules for disjunction and negation to a general conclusion, we can read off what the derived rules for implication should be.

$$\dfrac{\dfrac{\Gamma, A \;\#\; B, \Delta}{\Gamma \;\#\; \neg A, B, \Delta}\,\neg R}{\Gamma \;\#\; \neg A \vee B, \Delta}\,\vee R \qquad \text{gives us} \qquad \dfrac{\Gamma, A \;\#\; B, \Delta}{\Gamma \;\#\; A \supset B, \Delta}\,\supset R$$

$$\dfrac{\dfrac{\Gamma \;\#\; A, \Delta}{\Gamma, \neg A \;\#\; \Delta}\,\neg L \qquad \Gamma, B \;\#\; \Delta}{\Gamma, \neg A \vee B \;\#\; \Delta}\,\vee L \qquad \text{gives us} \qquad \dfrac{\Gamma \;\#\; A, \Delta \qquad \Gamma, B \;\#\; \Delta}{\Gamma, A \supset B \;\#\; \Delta}\,\supset L$$

These rules very much resemble the $\supset R$ and $\supset L$ rules from intuitionistic logic, though interestingly, the left rule does not require us to propagate the hypothesis $A \supset B$ into the left premise. ■

4 Proof Search in Prolog (20 points)

Refer to the classical sequent calculus of Problem 3.

Task 1 (20 pts). Complete the following Prolog code implementing a proof search strategy for our classical sequent calculus. You may use the predicate `intersect` from Problem 1.

Solution. We use the variable `G` to represent the context Γ, the variable `D` to represent the context Δ, and `Ps` and `Qs` to hold true and false hypotheses until we're ready to try the "init" rule. The code follows. ■

```
decide(Ps, Qs, G, [~ A | D]) :-
    decide(Ps, Qs, [A | G], D).

decide(Ps, Qs, [~ A | G], D) :-
    decide(Ps, Qs, G, [A | D]).

decide(Ps, Qs, G, [A \/ B | D]) :-

    decide(Ps, Qs, G, [A, B | D])
    ____________________________________________________________ .


decide(Ps, Qs, [A \/ B | G], D) :-

    decide(Ps, Qs, [A | G], D)
    ____________________________________________________________ ,

    decide(Ps, Qs, [B | G], D)
    ____________________________________________________________ .


decide(Ps, Qs, G, [? P | D]) :-

    decide(Ps, [? P | Qs], G, D)
    ____________________________________________________________ .


decide(Ps, Qs, [? P | G], D) :-

    decide([? P | Ps], Qs, G, D)
    ____________________________________________________________ .


decide(Ps, Qs, [], []) :-

    intersect(Ps, Qs)
    ____________________________________________________________ .
```

5 Representing Proofs in LF (15 points)

Recall that the logical framework LF offers a clean and elegant representation of sequent calculi with weakening and contraction in which sequent calculus hypotheses are represented as LF hypotheses. If we are interested in proving things about the calculus from Problem 3 rather than using it for proof search, such a representation is very convenient.

Task 1 (15 pts). Imagine we permit implicit contraction in our classical sequent calculus. Recall that we interpret a sequent Γ `#` Δ in the following way: if all the propositions in Γ are true and all the propositions in Δ are false, then they are in contradiction. We exploit this view in LF by representing every A in Γ as a hypothesis `true A`, and every B in Δ as a hypothesis `false B`.

Since every provable sequent represents a contradiction, the conclusion in LF is always a new judgment `contra`.

Complete the following LF representation of the new calculus using the ideas above.

```
contra : type.
true : prop -> type.
false : prop -> type.



init : true (? P) -> false (? P) -> contra
       ____________________________________________________________ .


~R : (true A -> contra)
  -> (false (~ A) -> contra).

~L : (false A -> contra)
  -> (true (~ A) -> contra).




\/R : (false A -> false B -> contra)
      ____________________________________________________________

   -> (false (A \/ B) -> contra)
      ____________________________________________________________ .


\/L : (true A -> contra)
   -> (true B -> contra)
   -> (true (A \/ B) -> contra).
```