Final Exam

15-317/657 Constructive Logic
André Platzer

December 17, 2015

Name: _____

Andrew ID: _____

Instructions

- This exam is open-book, closed internet.
- Remember to label all inference rules in your deductions.
- Throughout this exam, explain whenever there are notable steps or choices or subtleties and justify the rationale for your particular choice!
- You have 3 hours to complete the exam.
- There are 6 problems on 17 pages, including blank pages for extra space *at the end*.
- Consider writing out deductions on scratch paper first.

	Max	Score
Sequent Calculus	50	
Proof Checking	50	
Miraculous Sequent Rules	50	
Substitutions	60	
Unification	50	
Prolog	40	
Total:	300	

1 Sequent Calculus (50 points)

This question considers the sequent calculus with cut, weakening, and identity.

20 **Task 1** Prove the following equivalence using sequent calculus

$(A \wedge B) \supset (C \wedge D) \equiv A \supset ((B \supset C) \wedge (B \supset D))$

10 **Task 2** Prove the following theorem:
Theorem (Disconnection Property): *If* $\Longrightarrow (\neg A \vee B) \wedge C$ *then either* $A \Longrightarrow \bot$ *or* $\Longrightarrow B$ *and, either way, also* $\Longrightarrow C$.

20 **Task 3** Recall that $\neg\neg A \supset A$ is not provable in the (intuitionistic) sequent calculus. Give a simple proof that the law of excluded middle $A \vee \neg A$ is not provable in the (intuitionistic) sequent calculus either.

2 Proof Checking (50 points)

30 **Task 1** Commodore Horgiatiki performed one case of a cut elimination proof for sequent calculus. But he is missing some parts and is unsure whether he got a correct proof. Fill in literally **all** missing arguments and justifications and steps so that you obtain a complete proof. If there are any errors or missing justifications in Horgiatiki's proof, identify and clearly mark all errors and explain carefully **why** they are incorrect arguments.

If

$\Gamma \Longrightarrow A_1$		(1)
$\Gamma \Longrightarrow A_2$		(2)
$\Gamma \Longrightarrow A_1 \wedge A_2$	by $\wedge R$ on (1) and (2)	(3)
$\Gamma, A_1 \wedge A_2, A_2 \Longrightarrow C$		(4)
$\Gamma, A_1 \wedge A_2 \Longrightarrow C$	by $\wedge L_2$ on (4)	(5)

Then

$\Gamma, A_1 \Longrightarrow C$	by i.h. on _____ and (3) and (4)	(6)
$\Gamma, A_1 \wedge A_2 \Longrightarrow C$	by i.h. on _____ and (2) and (4)	(7)
$\Gamma \Longrightarrow C$	by i.h. on _____ and (3) and (7)	(8)

20 **Task 2** Mark all errors in the following sequent calculus proof and subsequently explain whether and why they are soundness-critical or why they could be accepted with a different argument.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{⑦}}{⑥\ a(x) \supset p(x,x) \Longrightarrow a(x) \supset p(x,x)}\ init}{⑤\ a(x) \supset p(x,x), a(x) \Longrightarrow p(x,x)}\ \supset R}{④\ a(x), a(x) \supset p(x,x) \Longrightarrow \forall y\, p(y,x)}\ \forall R}{③\ a(x), \forall x\, (a(x) \supset p(x,x)) \Longrightarrow \forall y\, p(y,x)}\ \supset L}{②\ \forall x\, (a(x) \supset p(x,x)) \Longrightarrow a(x) \supset \forall y\, p(y,x)}}{①\ \forall x\, (a(x) \supset p(x,x)) \Longrightarrow \forall y\, (a(y) \supset \forall x\, p(x,y))}\ \forall R}{⓪\ \Longrightarrow \forall x\, (a(x) \supset p(x,x)) \supset \forall y\, (a(y) \supset \forall x\, p(x,y))}\ \supset L$$

Refer to the line numbers in your answers below. Where ① refers to the whether the sequent in line ① can result by applying the given proof rule to the sequent in ⓪ etc.

⓪ acceptable as proof search starts with any well-formed sequent as initial conjecture.

①

②

③

④

⑤

⑥

⑦

3 Miraculous Sequent Rules (50 points)

In this question, we consider suggestions for new and improved proof rules that fierce Captain Toughch came up with. *Either* show the proof rules to be sound by deriving them or proving them to be admissible. *Or* show that they can be used to prove a formula that we cannot prove soundly and *explain briefly* why that formula should not be proved.

10 **Task 1**

$$\frac{\Gamma, A \vee D \Longrightarrow C}{\Gamma, A \supset B \Longrightarrow C \supset B}\ R1$$

Consider linear logic now.

10 **Task 2**

$$\frac{\Gamma; \cdot \Vdash A \multimap B}{\Gamma; \Delta \Vdash\ !(A \multimap B)}\ R2$$

10 **Task 3**

$$\frac{\Gamma; \Delta \Vdash A \quad \Gamma; \Delta \Vdash B \multimap C}{\Gamma; \Delta, A \multimap B \Vdash C}\ R3$$

10 **Task 4**

$$\frac{\Gamma; \Delta \Vdash A \quad \Gamma; \Delta \Vdash !(B \multimap C)}{\Gamma; \Delta, A \multimap B \Vdash C}\ R4$$

10 **Task 5** Recall natural deduction rules for intuitionistic propositional logic such as

$$\frac{B\ \mathit{true}}{A \vee B\ \mathit{true}}\ \vee I_R$$

Can you give such a natural deduction proof calculus for linear logic? Briefly justify why or why not.

4 Substitutions (60 points)

Recall that a *substitution* is a function σ from terms to terms that satisfies

$$f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma) \qquad \text{for all function symbols } f \text{ and terms } t_i$$

and has a finite domain $\mathrm{dom}(\sigma) = \{x \,:\, x\sigma \neq x\}$ of variables. Recall that $\tau\sigma$ denotes the substitution that is the composition of substitution σ after τ. Finally recall that a *variable renaming* is a substitution whose only effect is to replace variables by variables, not by arbitrary terms, and that, moreover, never renames two different variables to the same variable.

20 **Task 1** Let σ and τ be substitutions such that $\tau\sigma = (\cdot)$ is the identity substitution. Then is σ a variable renaming? Prove or disprove.

20 **Task 2** Let σ and τ be any substitutions such that $\tau\sigma = \tau$. Then is σ a variable renaming? Prove or disprove.

Recall that a *representation ℓ for a substitution* is of the form, e.g.:

$$\ell = (r_1/x_1, r_2/x_2, \ldots, r_n/x_n)$$

For any such representation ℓ of a substitution, let $\hat{\ell}$ denote the substitution belonging to that representation ℓ.

20 **Task 3** Let ℓ, k be representations of substitutions $\hat{\ell}$ and $\hat{k}$, respectively. Under what condition on ℓ and k is $\ell \cup k$ a representation of the composition $\hat{k}\hat{\ell}$ of $\hat{\ell}$ after $\hat{k}$? Prove correctness of the condition you identified or prove why no such condition exists.

5 Unification (50 points)

Unification specified the judgment $t \doteq s \mid \theta$ where θ is the most-general unifier for terms t and s. But, with some caveats, it works for formulas, too. In this question, we construct the judgment $F \doteq G \mid \theta$ with most-general unifier θ for formulas F and G.

10 **Task 1** Augment the judgment by writing new inference rules to also cover the case $p(\bar{t})$ for predicate symbol p with a sequence of terms $\bar{t}$ as arguments.

10 **Task 2** Augment the judgment further to the case of formulas of the form $F \vee G$.

10 **Task 3** Give a most-general unifier of

$$p(f(x), x) \vee q(g(u, x))$$

and

$$p(z, g(b, c)) \vee q(g(z, y))$$

20 **Task 4** Prove *soundness* of your answers from Tasks 1 and 2, i.e. that the result, θ, of $F \doteq G \mid \theta$ indeed is a unifier for formulas F and G.

6 Prolog (40 points)

In this question we will study ways of computing the derivative of polynomials in one variable with Prolog. Assume that polynomial expressions are represented as data structures of type `poly` built in an arbitrary shape from these constructors:

```
plus(S,T)  represents the sum of S and T
times(S,T) represents the product of S and T
var        indicates the variable (only one variable occurs so no need for a name)
num(N)     represents the number literal N (say as an integer)
```

In this problem you will define a predicate `diff/2` to compute the derivative of a polynomial expression represented in this way. For example, the following query is expected to succeed:

```
?- diff(plus(var,num(5)), plus(num(1), num(0))).
```

Modes in Prolog describe the intended ways of using a predicate. Mode `+poly` refers to an input argument of type `poly` that needs to be provided. Mode `-poly` refers to an output argument of type `poly` that will be computed by the predicate when all inputs are provided.

20 **Task 1** Write a Prolog program `diff(+poly,-poly)` that takes the polynomial as an input in the first argument and produces its derivative as an output in the second argument.

10 **Task 2** With mode `diff(+poly,-poly)`, the predicate from Task 1 computes a derivative. Is there a mode with which the predicate from Task 1 computes antiderivatives (also known as indefinite integrals)? Justify.

10 **Task 3** Is there a mode with which the predicate from Task 1 can be used to check whether a given polynomial expression is the integral of another given polynomial expression? Justify.

Blank page for extra answers if needed