Final Exam
15-816 Substructural Logics
Frank Pfenning
December 12, 2016

Name: Andrew ID:

Instructions

• This exam is closed-book, closed-notes.
• You have 3 hours to complete the exam.
• There are 6 problems.

<table>
<thead>
<tr>
<th></th>
<th>Ordered Logic</th>
<th>Focusing</th>
<th>Call by Value</th>
<th>Cost Semantics</th>
<th>SSOS</th>
<th>True Concurrency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prob</td>
<td>Prob 1</td>
<td>Prob 2</td>
<td>Prob 3</td>
<td>Prob 4</td>
<td>Prob 5</td>
<td>Prob 6</td>
</tr>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>45</td>
<td>45</td>
<td>60</td>
<td>40</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>
Problem 1: Ordered Logic (45 pts)

There is a “quick check” whether a sequent in the fragment of ordered logic with $A \setminus B$ and $A \cdot B$ may be provable by translating the sequent into the free group over its propositional variables and checking whether the antecedents and succedent denote the same group element.

Recall that a group can be defined by a binary operator $a \cdot b$, a unit element $e$, and an inverse operator $a^{-1}$ satisfying the laws on the left, with some additional useful properties on the right.

\[
\begin{align*}
(a \cdot b) \cdot c &= a \cdot (b \cdot c) \quad & (a^{-1})^{-1} &= a \\
a \cdot e &= a = e \cdot a \quad & e^{-1} &= e \\
a \cdot a^{-1} &= e = a^{-1} \cdot a \quad & (a \cdot b)^{-1} &= b^{-1} \cdot a^{-1}
\end{align*}
\]

The interpretation of propositions and antecedents is defined by

\[
\begin{align*}
[p] &= p & \text{for atoms or propositional variables } p \\
[A \cdot B] &= [A] \cdot [B] \\
[A \setminus B] &= [A]^{-1} \cdot [B] \\
[\ [] &= e \\
[\Omega_1 \Omega_2] &= [\Omega_1] \cdot [\Omega_2]
\end{align*}
\]

Then for any $A$ such that $\Omega \vdash A$ we have $[\Omega] = [A]$. For example, $\vdash a \setminus (b \setminus (b \cdot a))$ and

\[
[a \setminus (b \setminus (b \cdot a))] = a^{-1} \cdot [b \setminus (b \cdot a)] = a^{-1} \cdot b^{-1} \cdot [b \cdot a] = a^{-1} \cdot b^{-1} \cdot b \cdot a = a^{-1} \cdot a = e = [ ]
\]

Task 1 (5 pts). Apply this test to check whether

\[
((a \setminus b) \setminus (a \setminus a)) \setminus c \vdash (a \setminus a) \setminus ((b \setminus a) \setminus c)
\]

might be provable. Do not try to prove or refute this formula.
**Task 2** (5 pts). Find two propositions $A_0$ and $B_0$ consisting only of propositional variables and the connective \ such that $A_0 \vdash B_0$ passes the test but is not provable.

\[
A_0 = \\
B_0 = 
\]

**Task 3** (20 pts). Fill in some cases in the proof that $\Omega \vdash A$ implies $[\Omega] = [A]$.

**Proof**: By induction of the deduction of $\Omega \vdash A$.

**Case**: Rule id

**Case**: Rule $\backslash R$

**Case**: Rule $\backslash L$
**Task 4** (10 pts). Extend the translation to encompass $A / B$, $A \circ B$ and $1$ so that the test remains valid. You do not need to extend the proof.

\[
[A / B] =
\]

\[
[A \circ B] =
\]

\[
[1] =
\]

**Task 5** (5 pts). Explain how to adapt this test to multiplicative linear logic with connectives $A \multimap B$, $A \otimes B$, and $1$, and provide the interpretation of these connectives below.

\[
[A \multimap B] = [A]^{-1} \cdot [B]
\]

\[
[A \otimes B] = [A] \cdot [B]
\]

\[
[1] = e
\]
Problem 2: Focusing (45 pts)

Consider the sentence *John never works for Jane* where we attached the following types to the sentence constituents:

\[
\begin{array}{cccccc}
\text{John} & \text{never} & \text{works} & \text{for} & \text{Jane} \\
: & : & : & : & : \\
 n & (n \setminus s) \setminus (n \setminus s) & n \setminus s & (s \setminus s) / n & n \vdash s \\
\end{array}
\]

**Task 1** (5 pts). Assume \(n\) is positive and \(s\) is negative. Polarize the following definitions by inserting the minimal number of shifts.

\[
\begin{align*}
\text{adv} &= \ ( n \ \setminus \ s \ ) \setminus ( n \ \setminus \ s \ ) \\
\text{itv} &= \ n \ \setminus \ s \\
\text{prep} &= \ ( s \ \setminus \ s \ ) \setminus n \\
\end{align*}
\]

**Task 2** (20 pts). Provide all the synthetic rules of inference that arise from focusing on propositions in the sequent that represents parsing *John never works for Jane* as a sentence.
Task 3 (20 pts). Provide all the possible complete or partial failing proofs ending in

\[ n \text{ adv itv prep } n \vdash s \]

using only the synthetic rules of inference. We think of proof construction proceeding upwards, from the conclusion. Write out the (partial) proof below and fill in:

- There are _________ different complete proofs.
- There are _________ failing incomplete proofs.
Problem 3: Call-by-Push-Value (60 pts)

In this problem we explore call-by-push-value (CBPV)

Task 1 (5 pts). In CBPV, 

*computations* are (circle one) (i) positive or (ii) negative 

*values* are (circle one) (i) positive or (ii) negative 

Task 2 (20 pts). Annotate the given rules with their terms in CBPV on the right and give the types A and B their correct polarity. Use M, N to stand for computations and V, W for values.

\[
\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B \rightarrow I}
\]

\[
\frac{\Gamma \vdash A \rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B \rightarrow E}
\]

\[
\frac{\Gamma \vdash A}{\Gamma \vdash \downarrow A \downarrow I}
\]

\[
\frac{\Gamma \vdash \downarrow A}{\Gamma \vdash A \downarrow E}
\]

Task 3 (5 pts). Give the local reduction(s) for \(\rightarrow I\) followed by \(\rightarrow E\). You only need to express it on the proof terms, not the deductions.

Task 4 (5 pts). Give the local reduction(s) for \(\downarrow I\) followed \(\downarrow E\). You only need to express it on the proof terms, not the deductions.
Task 5 (5 pts). Polarize the following two types using only ↓, also assigning polarities to type variables $A, B,$ and $C$ in each case.

\[
A \rightarrow ( B \rightarrow A )
\]

\[
( A \rightarrow ( B \rightarrow C ) ) \rightarrow ( ( A \rightarrow B ) \rightarrow ( A \rightarrow C ) )
\]

Task 6 (5 pts). We write $\overline{E}$ for the translation of a simply-typed term $E$ into CBPV. Insert appropriate constructs so that the following simply-typed term is well-typed under CBPV and your polarization.

\[
K : A \rightarrow (B \rightarrow A)
\]

\[
= \lambda x. \lambda y. x
\]

\[
\overline{K} = \lambda x. \lambda y. \text{force } x
\]

Task 7 (5 pts). Insert appropriate constructs so that the following simply-typed terms well-typed under CBPV and your polarization.

\[
S : (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))
\]

\[
= \lambda x. \lambda y. \lambda z. (x z) (y z)
\]

\[
\overline{S} = \lambda x. \lambda y. \lambda z. ((\text{force } x) z) (\text{thunk } (\text{force } y) z))
\]

Task 8 (10 pts). Compute the terminal computation or value corresponding to the properly polarized form of $(S K) K$ by applying local reductions anywhere in the term. Show the result of each reduction.

\[
(\overline{S} \overline{K}) \overline{K} = \]

\[
\rightarrow
\]

\[
\vdash
\]
Problem 4: Cost Semantics (40 pts)

In this problem we consider the ordered substructural operational semantics for the subsingleton fragment of ordered logic with $\oplus$ and $1$.

Task 1 (20 pts). Complete the following rules to describe asynchronous communication. The first two rules have been filled in for you.

\[
\begin{align*}
&\frac{\text{proc}(P \mid Q)}{\text{proc}(P) \quad \text{proc}(Q)} & \frac{\text{proc}(\leftrightarrow)}{} \\
&. & . \\
\end{align*}
\]

Computation rules for $\oplus$ (process expressions $R.l_k : P$ and $(\text{case}_L (l_i \Rightarrow Q_i)_{i \in I})$)

Rules for $1$ (process expressions closeR and waitL ; $Q$)
Task 2 (20 pts). Instrument the operational semantics to count the total number of processes that are spawned. Assume we start with the configuration \( \text{proc}(1, P) \) for \( \cdot \vdash P : 1 \). If we terminate with the configuration \( \text{msg}(k, \text{closeR}) \) then \( k \) should be the total number of processes spawned during the computation. Do not count any messages. Feel free to substitute, add, or delete rules.
**Problem 5: Substructural Operational Semantics (40 pts)**

Consider the typing rules for the constructs in call-by-push-value associated with $\uparrow A^+$.

\[
\begin{align*}
\frac{}{\Gamma \vdash \text{return} \ V : \uparrow A^+} & \quad \uparrow I \\
\frac{\Gamma \vdash M : \uparrow A^+ \quad \Gamma, x: A^+ \vdash N : C^-}{\Gamma \vdash \text{let} \ val \ x = M \ \text{in} \ N : C^-} & \quad \uparrow E
\end{align*}
\]

We present the evaluation rules in the form of an ordered substructural operational semantics, which is based on three predicates $\text{eval}(M)$, $\text{retn}(T)$, and $\text{cont}(K)$, where $M$ is a computation, $T$ is a terminal computation, and $K$ is a continuation with a “hole” indicated by an underscore.

\[
\begin{align*}
ev_{\text{letval}} & : \text{eval}(\text{let} \ val \ x = M \ \text{in} \ N) \ \uparrow (\text{eval}(M) \ \bullet \ \text{cont}(\text{let} \ val \ x = \_ \ \text{in} \ N)) \\
ev_{\text{return}} & : \text{eval}(\text{return} \ V) \ \uparrow \ \text{retn}(\text{return} \ V) \\
r_{\text{return}} & : \text{retn}(\text{return} \ V) \ \bullet \ \text{cont}(\text{let} \ val \ x = \_ \ \text{in} \ N) \ \uparrow \ \text{eval}([V/x]N)
\end{align*}
\]

**Task 1 (20 pts).** Re-express the ordered specification in a linear framework such as CLF by adding destinations.

\[
\begin{align*}
ev_{\text{letval}} & : \\
ev_{\text{return}} & : \\
r_{\text{return}} & :
\end{align*}
\]
Task 2 (20 pts). Now we would like to introduce some parallelism into the evaluation of \texttt{let val x = M in N}. Informally, we evaluate $M$ and $N$ concurrently, with a new destination $d$ for $x$ acting as a form of channel connecting $M$ and $N$.

In the specification, you may need a different form of continuation, and revise and possibly add some rules. Introduce a new \textit{persistent} predicate $\texttt{bind}(V, d)$ which states that the value of the destination $d$ is permanently the value $V$. 
Problem 6: True Concurrency (20 pts)

Task 1 (10 pts). What is true concurrency?

Task 2 (10 pts). How is true concurrency manifest in the Concurrent Logical Framework (CLF)?
Appendix: Some Inference Rules

Propositions \( A, B, C \) ::= \( p \mid A \oplus B \mid A \& B \mid 1 \)
\( \mid A / B \mid B \setminus A \mid A \bullet B \mid A \circ B \)

Judgmental rules

\[
\begin{align*}
\frac{}{A \vdash A} & \quad \text{id}_A \\
\frac{\Omega \vdash A}{\Omega \vdash \Omega, A, \Omega_R \vdash C} & \quad \text{cut}_A \\
\end{align*}
\]

Propositional rules

\[
\begin{align*}
\frac{A \vdash B}{\Omega \vdash A \setminus B} & \quad \backslash R \\
\frac{\Omega \vdash A}{\Omega \vdash B / A} & \quad / R \\
\frac{\Omega \vdash A \quad \Omega' \vdash B}{\Omega \& \Omega' \vdash A \bullet B} & \quad \bullet R \\
\frac{\Omega \vdash B \quad \Omega' \vdash A}{\Omega \& \Omega' \vdash A \circ B} & \quad \circ R \\
\frac{\Omega \vdash A}{\Omega \vdash 1} & \quad \text{1R} \\
\frac{\Omega \vdash \Omega, A, \Omega_R \vdash C}{\Omega \vdash 1, \Omega_R \vdash C} & \quad \text{1L} \\
\frac{\Omega \vdash A}{\Omega \vdash A \oplus B} & \quad \oplus R_1 \\
\frac{\Omega \vdash B}{\Omega \vdash A \oplus B} & \quad \oplus R_2 \\
\frac{\Omega \vdash A \quad \Omega \vdash R}{\Omega \vdash A \oplus B} & \quad \oplus L \\
\frac{\Omega \vdash A \quad \Omega \vdash B}{\Omega \vdash A \& B} & \quad \& R \\
\frac{\Omega \vdash A \quad \Omega \vdash R}{\Omega \vdash (A \& B) \Omega_R \vdash C} & \quad \& L_1 \\
\frac{\Omega \vdash B \quad \Omega \vdash R}{\Omega \vdash (A \& B) \Omega_R \vdash C} & \quad \& L_2
\end{align*}
\]
Types

\[ A, B, C ::= \oplus \{ l_i : A_i \}_{i \in I} \mid & \{ l_i : A_i \}_{i \in I} \mid 1 \]
\[ A / B \mid B \setminus A \mid A \bullet B \mid A \circ B \]

Processes \( P, Q \)

\[ x \leftarrow y \]
\[ x \leftarrow P_x ; Q_x \]
\[ x.l_k ; P \mid \text{case } x \left( l_i \Rightarrow Q_i \right)_{i \in I} \]
\[ \text{close } x \mid \text{wait } x ; Q \]
\[ \text{send } x \left( x \leftarrow \text{recv } x ; Q_x \right) \]

Judgmental Rules

\[
\frac{\Omega \vdash P_x :: (x : A) \quad \Omega_L (x : A) \quad \Omega_R \vdash Q_x :: (z : C)}{\Omega, \Omega_L, \Omega_R \vdash (x \leftarrow P_x ; Q_x) :: (z : C)} \quad \text{cut}
\]
\[
\frac{y : A \vdash x \leftarrow y :: (x : A)}{\text{id}}
\]

Propositional Rules

\[
\frac{\Omega \vdash P :: (x : A_k) \quad \left( k \in I \right) \quad \oplus R_k}{\Omega \vdash (x.l_k ; P) :: (x : \oplus \{ l_i : A_i \}_{i \in I})}
\]
\[
\frac{\Omega \vdash P_i :: (x : A_i) \quad \left( \forall i \in I \right) \quad \& R}{\Omega \vdash \text{case } x \left( l_i \Rightarrow P_i \right)_{i \in I} :: (x : \{ l_i : A_i \}_{i \in I})}
\]
\[
\frac{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; Q_x \right) \quad \left( 1 \right) \quad 1 R}{\Omega \vdash \text{close } x :: (x : 1)}
\]
\[
\frac{\Omega \vdash P :: (x : B) \quad \left( w : A \right) \quad \oplus L}{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; P_y \right) :: (x : B \setminus B)}
\]
\[
\frac{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; P_y \right) :: (x : A \setminus B) \quad \& L^*}{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; P_y \right) :: (x : A \bullet B)}
\]
\[
\frac{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; P_y \right) :: (x : A \circ B) \quad \circ L^*}{\Omega \vdash \text{send } x \left( x \leftarrow \text{recv } x ; P_y \right) :: (x : A \circ B)}
\]

Computation Rules

\[
\frac{\text{proc}(z, x \leftarrow P_x ; Q_x) \quad \text{cmp}\nu}{\text{proc}(w, P_w) \quad \text{proc}(z, Q_w)}
\]
\[
\frac{\text{proc}(x, x \leftarrow y)}{x = y \quad \text{fwd} \quad \text{proc}(x, \text{close } x) \quad \text{proc}(z, \text{wait } x ; Q) \quad \text{proc}(z, Q)} \quad 1 C
\]
\[
\frac{\text{proc}(x, x.l_k ; P) \quad \text{proc}(z, \text{case } x \left( l_i \Rightarrow Q_i \right)_{i \in I})}{\text{proc}(x, P) \quad \text{proc}(z, Q_k)} \quad \& C
\]
\[
\frac{\text{proc}(x, y \leftarrow \text{recv } x ; P_y) \quad \text{proc}(z, \text{send } x \leftarrow w ; Q)}{\text{proc}(x, P_y) \quad \text{proc}(z, Q) \quad / C \setminus C}
\]
\[
\frac{\text{proc}(x, x.l_k ; Q) \quad \text{proc}(z, x.l_k ; Q)}{\text{proc}(x, P) \quad \text{proc}(z, Q) \quad \& C}
\]
\[
\frac{\text{proc}(x, \text{send } x \leftarrow w ; P) \quad \text{proc}(z, y \leftarrow \text{recv } x ; Q_y) \quad \bullet C \setminus C}{\text{proc}(P) \quad \text{proc}(Q_w) \quad \bullet C, \circ C}
\]