Final Exam

15-816 Substructural Logics
Frank Pfenning

December 12, 2016

Name: **Sample Solution** Andrew ID: **fp**

Instructions

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

	Ordered Logic	Focusing	Call by Push Value	Cost Semantics	SSOS	True Concurrency	
	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Prob 6	Total
Score	**45**	**45**	**60**	**40**	**40**	**20**	**250**
Max	45	45	60	40	40	20	250

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 $A \bullet B$ *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.

$$\begin{array}{ll} (a \cdot b) \cdot c = a \cdot (b \cdot c) & (a^{-1})^{-1} = a \\ a \cdot e = a = e \cdot a & e^{-1} = e \\ a \cdot a^{-1} = e = a^{-1} \cdot a & (a \cdot b)^{-1} = b^{-1} \cdot a^{-1} \end{array}$$

The interpretation of propositions and antecedents is defined by

$$\begin{array}{lcll} [\![p]\!] & = & p & \text{for atoms or propositional variables } p \\ [\![A \bullet B]\!] & = & [\![A]\!] \cdot [\![B]\!] & \\ [\![A \setminus B]\!] & = & [\![A]\!]^{-1} \cdot [\![B]\!] & \\ & & & \\ [\![\ ]\!] & = & e & \\ [\![\Omega_1\ \Omega_2]\!] & = & [\![\Omega_1]\!] \cdot [\![\Omega_2]\!] & \end{array}$$

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

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

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.

$$\begin{array}{lllll} & [\![((a \setminus b) \setminus (a \setminus a)) \setminus c]\!] & \quad & & [\![(a \setminus a) \setminus ((b \setminus a) \setminus c)]\!] \\ = & [\![(a \setminus b) \setminus (a \setminus a)]\!]^{-1} \cdot c & & = & [\![a \setminus a]\!]^{-1} \cdot [\![(b \setminus a) \setminus c]\!] \\ = & ([\![a \setminus b]\!]^{-1} \cdot [\![a \setminus a]\!])^{-1} \cdot c & & = & (a^{-1} \cdot a)^{-1} \cdot [\![b \setminus a]\!]^{-1} \cdot c \\ = & ((a^{-1} \cdot b)^{-1} \cdot (a^{-1} \cdot a))^{-1} \cdot c & & = & (b^{-1} \cdot a)^{-1} \cdot c \\ = & (b^{-1} \cdot a \cdot a^{-1} \cdot a)^{-1} \cdot c & & = & a^{-1} \cdot b \cdot c \\ = & a^{-1} \cdot b \cdot c & & & \end{array}$$

Yes, they are equal! The sequent may be provable.

Task 2 (5 pts). Find two propositions A_0 and B_0 consisting only of propositional variables and the connective $\setminus$ such that $A_0 \vdash B_0$ passes the test but is not provable.

$$A_0 \quad = \quad a \setminus a$$

$$B_0 \quad = \quad b \setminus b$$

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

$$\frac{}{A \vdash A}\ \mathsf{id}$$

Then $[\![\Omega]\!] = [\![A]\!] = [\![A]\!]$.

Case: Rule $\backslash R$

$$\frac{A\ \Omega \vdash B}{\Omega \vdash A \setminus B}\ \backslash R$$

Then $[\![A]\!] \cdot [\![\Omega]\!] = [\![B]\!]$ by induction hypothesis. Multiply both sides by $[\![A]\!]^{-1}$ to obtain $[\![\Omega]\!] = [\![A]\!]^{-1} \cdot [\![B]\!] = [\![A \setminus B]\!]$

Case: Rule $\backslash L$

$$\frac{\Omega' \vdash A \quad \Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ \Omega'\ (A \setminus B)\ \Omega_R \vdash C}\ \backslash L$$

Then $[\![\Omega']\!] = [\![A]\!]$ and $[\![\Omega_L]\!] \cdot [\![B]\!] \cdot [\![\Omega_R]\!] = [\![C]\!]$ by induction hypothesis.
Now $[\![\Omega_L]\!] \cdot [\![\Omega']\!] \cdot ([\![A]\!]^{-1} \cdot [\![B]\!]) \cdot [\![\Omega_R]\!] = [\![\Omega_L]\!] \cdot [\![A]\!] \cdot [\![A]\!]^{-1} \cdot [\![B]\!] \cdot [\![\Omega_R]\!] = [\![C]\!]$.

Task 4 (10 pts). Extend the translation to encompass $A \,/\, B$, $A \circ B$ and $\mathbf{1}$ so that the test remains valid. You do not need to extend the proof.

$$[\![A \,/\, B]\!] = [\![A]\!] \cdot [\![B]\!]^{-1}$$

$$[\![A \circ B]\!] = [\![B]\!] \cdot [\![A]\!]$$

$$[\![\mathbf{1}]\!] = e$$

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

We add *commutativity* to the laws, $a \cdot b = b \cdot a$, so that the interpretation is into the free Abelian group over the propositional variables.

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

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

$$[\![\mathbf{1}]\!] = e$$

Problem 2: Focusing (45 pts)

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

$$\begin{array}{cccccc} \textit{John} & \textit{never} & \textit{works} & \textit{for} & \textit{Jane} & \\ : & : & : & : & : & \\ n & (n \backslash s) / (n \backslash s) & n \backslash s & (s \backslash s) / n & n & \vdash s \end{array}$$

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

$$\begin{array}{lcl} \mathsf{adv}^- & = & (\ n^+\ \backslash\ s^-\ )\ /\ \downarrow(\ n^+\ \backslash\ s^-\ ) \\ \mathsf{itv}^- & = & n^+\ \backslash\ s^- \\ \mathsf{prep}^- & = & (\downarrow s^-\ \backslash\ s^-\ )\ /\ n^+ \end{array}$$

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.

$$\dfrac{\dfrac{\dfrac{n^+\ \Omega_{21} \vdash s^-}{\Omega_{21} \vdash n^+ \backslash s^-}\backslash R}{\Omega_{21} \vdash [\downarrow(n^+ \backslash s^-)]}\downarrow R \quad \dfrac{\dfrac{\Omega_{12} = n^+}{\Omega_{12} \vdash [n^+]}\mathsf{id}^+ \quad \dfrac{\Omega_{11} = \Omega_{22} = \cdot; C = s^-}{\Omega_{11}\ [s^-]\ \Omega_{22} \vdash C}\mathsf{id}^-}{\Omega_1\ [n^+ \backslash s^-]\ \Omega_{22} \vdash C}\backslash R;\ \Omega_1 = \Omega_{11}\ \Omega_{12}}{\Omega_1\ [\mathsf{adv}^-]\ \Omega_2 \vdash C}/R;\ \Omega_2 = \Omega_{21}\ \Omega_{22} \qquad \dfrac{n^+\ \Omega \vdash s^-}{n^+\ \mathsf{adv}^-\ \Omega \vdash s^-}\mathsf{ADV}$$

$$\dfrac{\dfrac{\Omega_{12} = n^+}{\Omega_{12} \vdash [n^+]}\mathsf{id}^+ \quad \dfrac{\Omega_{11} = \Omega_2 = \cdot; C = s^-}{\Omega_{11}\ [s^-]\ \Omega_2 \vdash C}\mathsf{id}^-}{\Omega_1\ [\mathsf{itv}^-]\ \Omega_2 \vdash C}\backslash R;\ \Omega_1 = \Omega_{11}\ \Omega_{12} \qquad \dfrac{}{n^+\ \mathsf{itv}^- \vdash s^-}\mathsf{ITV}$$

$$\dfrac{\dfrac{\Omega_{21} = n^+}{\Omega_{21} \vdash [n^+]}\mathsf{id}^+ \quad \dfrac{\dfrac{\Omega_{12} \vdash s^-}{\Omega_{12} \vdash [\downarrow s^-]}\downarrow R \quad \dfrac{\Omega_{11} = \Omega_{22} = \cdot; C = s^-}{\Omega_{11}\ [s^-]\ \Omega_{22} \vdash C}\mathsf{id}^-}{\Omega_1\ [\downarrow s^- \backslash s^-]\ \Omega_{22} \vdash C}\backslash R;\ \Omega_1 = \Omega_{11}\ \Omega_{12}}{\Omega_1\ [\mathsf{prep}^-]\ \Omega_2 \vdash C}/R;\ \Omega_2 = \Omega_{21}\ \Omega_{22} \qquad \dfrac{\Omega \vdash s^-}{\Omega\ \mathsf{prep}^-\ n^+ \vdash s^-}\mathsf{PREP}$$

Task 3 (20 pts). Provide all the possible complete or partial failing proofs ending in

$$n \quad \mathsf{adv} \quad \mathsf{itv} \quad \mathsf{prep} \quad n \quad \vdash \quad s$$

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 ____2____ different complete proofs.
- There are ____0____ failing incomplete proofs.

Initially, only two rules apply: ADV or PREP. After that first step, all the steps are forced. We only count situations where a rule can proceed at least for one step, which is why our answer above is 0 (there are other correct answers, depending on the more precise definition).

$$\dfrac{\dfrac{\dfrac{}{n\ \mathsf{itv} \vdash s}\ \mathsf{ITV}}{n\ \mathsf{itv}\ \mathsf{prep}\ n \vdash s}\ \mathsf{PREP}}{n\ \mathsf{adv}\ \mathsf{itv}\ \mathsf{prep}\ n\ \vdash\ s}\ \mathsf{ADV} \qquad \dfrac{\dfrac{\dfrac{}{n\ \mathsf{itv} \vdash s}\ \mathsf{ITV}}{n\ \mathsf{adv}\ \mathsf{itv} \vdash s}\ \mathsf{ADV}}{n\ \mathsf{adv}\ \mathsf{itv}\ \mathsf{prep}\ n\ \vdash\ s}\ \mathsf{PREP}$$

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	NEGATIVE
values are (circle one)	(i) positive	or (ii) negative	POSITIVE

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.

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \to I \qquad \frac{\Gamma, x{:}A^+ \vdash M : B^-}{\Gamma \vdash \lambda x.\, M : A^+ \to B^-} \to I$$

$$\frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B} \to E \qquad \frac{\Gamma \vdash M : A^+ \to B^- \quad \Gamma \vdash V : A^+}{\Gamma \vdash M\ V : B^-} \to E$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash {\downarrow}A} {\downarrow}I \qquad \frac{\Gamma \vdash M : A^-}{\Gamma \vdash \mathsf{thunk}\ M : {\downarrow}A^-} {\downarrow}I$$

$$\frac{\Gamma \vdash {\downarrow}A}{\Gamma \vdash A} {\downarrow}E \qquad \frac{\Gamma \vdash V : {\downarrow}A^-}{\Gamma \vdash \mathsf{force}\ V : A^-} {\downarrow}E$$

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

$$(\lambda x.\, M)\, V \longrightarrow [V/x]M$$

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.

$$\mathsf{force}\ (\mathsf{thunk}\ M) \longrightarrow M$$

Task 5 (5 pts). Polarize the following two types using only $\downarrow$, also assigning polarities to type variables A, B, and C in each case.

Other polarizations are possible; here is one where all propositional variables are negative.

$$\downarrow A^- \;\rightarrow\; (\downarrow B^- \;\rightarrow\; A^-\;)$$

$$\downarrow(\downarrow A^- \;\rightarrow\; (\downarrow B^- \;\rightarrow\; C^-\;)\;) \;\rightarrow\; (\downarrow(\downarrow A^- \;\rightarrow\; B^-\;) \;\rightarrow\; (\downarrow A^- \;\rightarrow\; C^-\;)\;)$$

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.

$$\begin{aligned} K &: A \to (B \to A) \\ &= \lambda x.\, \lambda y.\, x \end{aligned}$$

$$\overline{K} = \lambda x.\, \lambda y.\, \mathsf{force}\ x$$

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

$$\begin{aligned} S &: (A \to (B \to C)) \to ((A \to B) \to (A \to C)) \\ &= \lambda x.\, \lambda y.\, \lambda z.\, (x\ z)\ (y\ z) \end{aligned}$$

$$\overline{S} = \lambda x.\, \lambda y.\, \lambda z.\, ((\mathsf{force}\ x)\ z)\ (\mathsf{thunk}\ ((\mathsf{force}\ y)\ z))$$

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.

$$\begin{aligned} \overline{(S\ K)\ K} &= \overline{S}\ (\mathsf{thunk}\ \overline{K})\ (\mathsf{thunk}\ \overline{K}) \\ &\longrightarrow \lambda y.\, \lambda z.\, ((\mathsf{force}\ (\mathsf{thunk}\ \overline{K}))\ z)\ (\mathsf{thunk}\ ((\mathsf{force}\ y)\ z)) \\ &\longrightarrow \lambda z.\, ((\mathsf{force}\ (\mathsf{thunk}\ \overline{K}))\ z)\ (\mathsf{thunk}\ ((\mathsf{force}\ (\mathsf{thunk}\ \overline{K}))\ z)) \end{aligned}$$

at this point we have a terminal computation, but to see what we have ...

$$\begin{aligned} &\longrightarrow \lambda z.\, (\overline{K}\ z)\ (\mathsf{thunk}\ (\overline{K}\ z)) \\ &\longrightarrow \lambda z.\, (\lambda y.\, \mathsf{force}\ z)\ (\mathsf{thunk}\ (\overline{K}\ z)) \\ &\longrightarrow \lambda z.\, \mathsf{force}\ z \end{aligned}$$

Problem 4: Cost Semantics (40 pts)

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

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

$$\frac{\mathsf{proc}(P \mid Q)}{\mathsf{proc}(P) \quad \mathsf{proc}(Q)} \qquad \frac{\mathsf{proc}(\leftrightarrow)}{\cdot}$$

Computation rules for $\oplus$ (process expressions $\mathsf{R}.l_k \; ; \; P$ and $(\mathsf{caseL} \; (l_i \Rightarrow Q_i)_{i \in I})$)

$$\frac{\mathsf{proc}(\mathsf{R}.l_k \; ; \; P)}{\mathsf{proc}(P) \quad \mathsf{msg}(\mathsf{R}.l_k \; ; \leftrightarrow)} \qquad \frac{\mathsf{msg}(\mathsf{R}.l_k \; ; \leftrightarrow) \quad \mathsf{proc}(\mathsf{caseL} \; (l_i \Rightarrow Q_i)_{i \in I})}{\mathsf{proc}(Q_k)}$$

Rules for $\mathbf{1}$ (process expressions closeR and $\mathsf{waitL} \; ; \; Q$)

$$\frac{\mathsf{proc}(\mathsf{closeR})}{\mathsf{msg}(\mathsf{closeR})} \qquad \frac{\mathsf{msg}(\mathsf{closeR}) \quad \mathsf{proc}(\mathsf{waitL} \; ; \; Q)}{\mathsf{proc}(Q)}$$

Task 2 (20 pts). Instrument the operational semantics to count the total number of processes that are spawned. Assume we start with the configuration $\mathsf{proc}(1, P)$ for $\cdot \vdash P : \mathbf{1}$. If we terminate with the configuration $\mathsf{msg}(k, \mathsf{closeR})$ 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.

$$\frac{\mathsf{proc}(k, P \mid Q)}{\mathsf{proc}(1, P) \quad \mathsf{proc}(k, Q)}\ \mathsf{spawn} \qquad \frac{\mathsf{msg}(k', P) \quad \mathsf{proc}(k, \leftrightarrow)}{\mathsf{msg}(k' + k, P)}\ \mathsf{forward}^{+}$$

$$\frac{\mathsf{proc}(k, \mathsf{R}.l_k \ ; \ P)}{\mathsf{proc}(k, P) \quad \mathsf{msg}(0, \mathsf{R}.l_k \ ; \leftrightarrow)}\ \oplus s \qquad \frac{\mathsf{msg}(0, \mathsf{R}.l_k) \quad \mathsf{proc}(k, \mathsf{caseL}\ (l_i \Rightarrow Q_i)_{i \in I})}{\mathsf{proc}(k, Q_k)}\ \oplus r$$

$$\frac{\mathsf{proc}(k, \mathsf{closeR})}{\mathsf{msg}(k, \mathsf{closeR})}\ \mathbf{1}s \qquad \frac{\mathsf{msg}(k, \mathsf{closeR}) \quad \mathsf{proc}(k', \mathsf{waitL} \ ; \ Q)}{\mathsf{proc}(k + k', Q)}\ \mathbf{1}r$$

If we had negative connectives, we should also have

$$\frac{\mathsf{proc}(k', \leftrightarrow) \quad \mathsf{msg}(k, P)}{\mathsf{msg}(k' + k, P)}\ \mathsf{forward}^{-}$$

but there are other ways to solve the counting problem for forwarding.

Problem 5: Substructural Operational Semantics (40 pts)

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

$$\frac{\Gamma \vdash V : A^+}{\Gamma \vdash \mathsf{return}\, V : \uparrow A^+} \uparrow I \qquad \frac{\Gamma \vdash M : \uparrow A^+ \quad \Gamma, x{:}A^+ \vdash N : C^-}{\Gamma \vdash \mathsf{let\,val}\, x = M \,\mathsf{in}\, N : C^-} \uparrow E$$

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

$$\begin{array}{lcl} \mathsf{ev\_letval} & : & \mathsf{eval}(\mathsf{let\,val}\, x = M \,\mathsf{in}\, N) \setminus \uparrow (\mathsf{eval}(M) \bullet \mathsf{cont}(\mathsf{let\,val}\, x = \_ \,\mathsf{in}\, N)) \\ \mathsf{ev\_return} & : & \mathsf{eval}(\mathsf{return}\, V) \setminus \uparrow \mathsf{retn}(\mathsf{return}\, V) \\ \mathsf{rt\_return} & : & \mathsf{retn}(\mathsf{return}\, V) \bullet \mathsf{cont}(\mathsf{let\,val}\, x = \_ \,\mathsf{in}\, N) \setminus \uparrow \mathsf{eval}([V/x]N) \end{array}$$

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

$$\begin{array}{lcl} \mathsf{ev\_letval} & : & \mathsf{eval}(\mathsf{let\,val}\, x = M \,\mathsf{in}\, N, D) \\ & & \multimap \uparrow (\exists d'.\, \mathsf{eval}(M, d') \otimes \mathsf{cont}(d', \mathsf{let\,val}\, x = \_ \,\mathsf{in}\, N, D)) \\ \mathsf{ev\_return} & : & \mathsf{eval}(\mathsf{return}\ V, D') \multimap \uparrow \mathsf{retn}(\mathsf{return}\ V, D') \\ \mathsf{rt\_return} & : & \mathsf{retn}(\mathsf{return}\ V, D') \otimes \mathsf{cont}(D', \mathsf{let\,val}\, x = \_ \,\mathsf{in}\, N, D) \multimap \uparrow \mathsf{eval}([V/x]N, D) \end{array}$$

Task 2 (20 pts). Now we would like to introduce some parallelism into the evaluation of $\mathsf{let\,val}\, x = M \,\mathsf{in}\, N$. 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 *persistent* predicate $\underline{\mathsf{bind}}(V, d)$ which states that the value of the destination d is permanently the value V.

We introduce a $\mathsf{val}\, d$, a *value* that refers to a destination d. In the rule for evaluation of letval we substitute a new $\mathsf{val}\, d'$ it for the value variable x.

$$\mathsf{ev\_letval} \;:\; \mathsf{eval}(\mathsf{let\,val}\, x = M \,\mathsf{in}\, N, D) \multimap \uparrow (\exists d'.\, \mathsf{eval}(M, d') \otimes \mathsf{eval}([\mathsf{val}\, d'/x]N, D))$$

$$\mathsf{ev\_return} \;:\; \mathsf{eval}(\mathsf{return}\ V, D') \multimap \uparrow \underline{\mathsf{bind}}(V, D')$$

Now we need to update the rules that depend on the shape of a value to dereference in case they see a value destination. Here is one possible way to accomplish this, using the example of the force construct.

$$\mathsf{ev\_force} \;:\; \mathsf{eval}(\mathsf{force}\ (\mathsf{thunk}\ M), D) \multimap \uparrow \mathsf{eval}(M, D)$$

$$\mathsf{ev\_force\_val} \;:\; \mathsf{eval}(\mathsf{force}\ (\mathsf{val}\ D'), D) \otimes \underline{\mathsf{bind}}(V, D') \multimap \uparrow \mathsf{eval}(\mathsf{force}\ V, D)$$

Problem 6: True Concurrency (20 pts)

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

We say a semantics is *truly concurrent* if there is no way to observe the relative order of independent events.

Task 2 (10 pts). How is *true concurrency* manifest in the Concurrent Logical Framework (CLF)?

In CLF, steps in the computation are represented by $p = R$, where R is a term consuming antecedents describing the state of the computation, and p is a pattern binding variables that name new components of the state. Two events $p = R$ and $q = S$ are *independent* if no variables in p are used in S and no variables in q are used in R. Then the expressions

$$(\mathsf{let\ val}\ p = R\ \mathsf{in\ let\ val}\ q = S\ \mathsf{in}\ E) = (\mathsf{let\ val}\ q = S\ \mathsf{in\ let\ val}\ p = R\ \mathsf{in}\ E)$$

are equal and therefore indistinguishable in the framework.

Appendix: Some Inference Rules

$$\begin{array}{llll} \text{Propositions} & A, B, C & ::= & p \mid A \oplus B \mid A \mathbin{\&} B \mid \mathbf{1} \\ & & \mid & A \,/\, B \mid B \setminus A \mid A \bullet B \mid A \circ B \end{array}$$

Judgmental rules

$$\frac{}{A \vdash A}\ \mathsf{id}_A \qquad \frac{\Omega \vdash A \quad \Omega_L\ A\ \Omega_R \vdash C}{\Omega_L\ \Omega\ \Omega_R \vdash C}\ \mathsf{cut}_A$$

Propositional rules

$$\frac{A\ \Omega \vdash B}{\Omega \vdash A \setminus B}\ \setminus R \qquad \frac{\Omega' \vdash A \quad \Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ \Omega'\ (A \setminus B)\ \Omega_R \vdash C}\ \setminus L$$

$$\frac{\Omega\ A \vdash B}{\Omega \vdash B \,/\, A}\ /R \qquad \frac{\Omega' \vdash A \quad \Omega_L\ B\ \Omega_r \vdash C}{\Omega_L\ (B \,/\, A)\ \Omega'\ \Omega_R \vdash C}\ /L$$

$$\frac{\Omega \vdash A \quad \Omega' \vdash B}{\Omega\ \Omega' \vdash A \bullet B}\ \bullet R \qquad \frac{\Omega_L\ A\ B\ \Omega_R \vdash C}{\Omega_L\ (A \bullet B)\ \Omega_R \vdash C}\ \bullet L$$

$$\frac{\Omega \vdash B \quad \Omega' \vdash A}{\Omega\ \Omega' \vdash A \circ B}\ \circ R \qquad \frac{\Omega_L\ B\ A\ \Omega_R \vdash C}{\Omega_L\ (A \circ B)\ \Omega_R \vdash C}\ \circ L$$

$$\frac{}{\cdot \vdash \mathbf{1}}\ \mathbf{1}R \qquad \frac{\Omega_L\ \Omega_R \vdash C}{\Omega_L\ \mathbf{1}\ \Omega_R \vdash C}\ \mathbf{1}L$$

$$\frac{\Omega \vdash A}{\Omega \vdash A \oplus B}\ \oplus R_1 \quad \frac{\Omega \vdash B}{\Omega \vdash A \oplus B}\ \oplus R_2 \qquad \frac{\Omega_L\ A\ \Omega_R \vdash C \quad \Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ (A \oplus B)\ \Omega_R \vdash C}\ \oplus L$$

$$\frac{\Omega \vdash A \quad \Omega \vdash B}{\Omega \vdash A \mathbin{\&} B}\ \& R \qquad \frac{\Omega_L\ A\ \Omega_R \vdash C}{\Omega_L\ (A \mathbin{\&} B)\ \Omega_R \vdash C}\ \& L_1 \quad \frac{\Omega_L\ B\ \Omega_R \vdash C}{\Omega_L\ (A \mathbin{\&} B)\ \Omega_R \vdash C}\ \& L_2$$

$$
\begin{array}{llcll}
\text{Types} & A, B, C & ::= & \oplus\{l_i : A_i\}_{i \in I} \mid \&\{l_i : A_i\}_{i \in I} \mid \mathbf{1} & \\
& & \mid & A \,/\, B \mid B \setminus A \mid A \bullet B \mid A \circ B & \\
\\
\text{Processes} & P, Q & ::= & x \leftarrow y & \text{identity/forward} \\
& & \mid & x \leftarrow P_x \;;\; Q_x & \text{cut/spawn} \\
& & \mid & x.l_k \;;\; P \mid \mathsf{case}\; x\; (l_i \Rightarrow Q_i)_{i \in I} & \oplus, \& \\
& & \mid & \mathsf{close}\; x \mid \mathsf{wait}\; x \;;\; Q & \mathbf{1} \\
& & \mid & \mathsf{send}\; x\; y \;;\; P \mid y \leftarrow \mathsf{recv}\; x \;;\; Q_x & /, \setminus, \bullet, \circ
\end{array}
$$

Judgmental Rules

$$
\frac{\Omega \vdash P_x :: (x{:}A) \quad \Omega_L\; (x{:}A)\; \Omega_R \vdash Q_x :: (z{:}C)}{\Omega_L\; \Omega\; \Omega_R \vdash (x \leftarrow P_x \;;\; Q_x) :: (z{:}C)}\;\mathsf{cut}
\qquad
\frac{}{y{:}A \vdash x \leftarrow y :: (x{:}A)}\;\mathsf{id}
$$

Propositional Rules

$$
\frac{\Omega \vdash P :: (x{:}A_k) \quad (k \in I)}{\Omega \vdash (x.l_k \;;\; P) :: (x : \oplus\{l_i{:}A_i\}_{i \in I})}\;\oplus R_k
\qquad
\frac{\Omega_L\; (x{:}A_i)\; \Omega_R \vdash Q_i :: (z{:}C) \quad (\forall i \in I)}{\Omega_L\; (x{:}\oplus\{l_i{:}A_i\}_{i \in I})\; \Omega_R \vdash \mathsf{case}\; x\; (l_i \Rightarrow Q_i)_{i \in I} :: (z{:}C)}\;\oplus L
$$

$$
\frac{\Omega \vdash P_i :: (x{:}A_i) \quad (\forall i \in I)}{\Omega \vdash \mathsf{case}\; x\; (l_i \Rightarrow P_i)_{i \in I} :: (x{:}\&\{l_i{:}A_i\}_{i \in I}))}\;\& R
\qquad
\frac{\Omega_L\; (x{:}A_k)\; \Omega_R \vdash P :: (z{:}C) \quad (k \in I)}{\Omega_L\; (x : \&\{l_i{:}A_i\}_{i \in I})\; \Omega_R \vdash (x.l_k \;;\; Q) :: (z{:}C)}\;\& L_k
$$

$$
\frac{}{\cdot \vdash \mathsf{close}\; x :: (x{:}\mathbf{1})}\;\mathbf{1}R
\qquad
\frac{\Omega_L\; \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\; (x{:}\mathbf{1})\; \Omega_R \vdash (\mathsf{wait}\; x \;;\; Q) :: (z{:}C)}\;\mathbf{1}L
$$

$$
\frac{\Omega\; (y{:}A) \vdash P_y :: (x{:}B)}{\Omega \vdash (y \leftarrow \mathsf{recv}\; x \;;\; P_y) :: (x{:}B \,/\, A)}\;/R
\qquad
\frac{\Omega_L\; (x{:}B)\; \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\; (x{:}B \,/\, A)\; (w{:}A)\; \Omega_R \vdash (\mathsf{send}\; x\; w \;;\; Q) :: (z{:}C)}\;/L^*
$$

$$
\frac{(y{:}A)\; \Omega \vdash P_y :: (x{:}B)}{\Omega \vdash (y \leftarrow \mathsf{recv}\; x \;;\; P_y) :: (x{:}A \setminus B)}\;\setminus R
\qquad
\frac{\Omega_L\; (x{:}B)\; \Omega_R \vdash Q :: (z{:}C)}{\Omega_L\; (w{:}A)\; (x{:}A \setminus B)\; \Omega_R \vdash (\mathsf{send}\; x\; w \;;\; Q) :: (z{:}C)}\;\setminus L^*
$$

$$
\frac{\Omega \vdash P :: (x{:}B)}{(w{:}A)\; \Omega \vdash (\mathsf{send}\; x\; w \;;\; P) :: (x{:}A \bullet B)}\;\bullet R^*
\qquad
\frac{\Omega_L\; (y{:}A)\; (x{:}B)\; \Omega_R \vdash Q_y :: (z{:}C)}{\Omega_L\; (x{:}A \bullet B)\; \Omega_R \vdash (y \leftarrow \mathsf{recv}\; x \;;\; Q_y) :: (z{:}C)}\;\bullet L
$$

$$
\frac{\Omega \vdash P :: (x{:}B)}{\Omega\; (w{:}A) \vdash (\mathsf{send}\; x\; w \;;\; P) :: (x{:}A \circ B)}\;\circ R^*
\qquad
\frac{\Omega_L\; (x{:}B)\; (y{:}A)\; \Omega_R \vdash Q_y :: (z{:}C)}{\Omega_L\; (x{:}A \bullet B)\; \Omega_R \vdash (y \leftarrow \mathsf{recv}\; x \;;\; Q_y) :: (z{:}C)}\;\circ L
$$

Computation Rules

$$
\frac{\mathsf{proc}(z, x \leftarrow P_x \;;\; Q_x)}{\mathsf{proc}(w, P_w) \quad \mathsf{proc}(z, Q_w)}\;\mathsf{cmp}^w
\qquad
\frac{\mathsf{proc}(x, x \leftarrow y)}{x = y}\;\mathsf{fwd}
\qquad
\frac{\mathsf{proc}(x, \mathsf{close}\; x) \quad \mathsf{proc}(z, \mathsf{wait}\; x \;;\; Q)}{\mathsf{proc}(z, Q)}\;\mathbf{1}C
$$

$$
\frac{\mathsf{proc}(x, x.l_k \;;\; P) \quad \mathsf{proc}(z, \mathsf{case}\; x\; (l_i \Rightarrow Q_i)_{i \in I})}{\mathsf{proc}(x, P) \quad \mathsf{proc}(z, Q_k)}\;\oplus C
\qquad
\frac{\mathsf{proc}(x, \mathsf{case}\; x\; (l_i \Rightarrow P_i)_{i \in I}) \quad \mathsf{proc}(z, x.l_k \;;\; Q)}{\mathsf{proc}(x, Q) \quad \mathsf{proc}(z, P_k)}\;\& C
$$

$$
\frac{\mathsf{proc}(x, y \leftarrow \mathsf{recv}\; x \;;\; P_y) \quad \mathsf{proc}(z, \mathsf{send}\; x\; w \;;\; Q)}{\mathsf{proc}(x, P_w) \quad \mathsf{proc}(z, Q)}\;/C, \setminus C
\qquad
\frac{\mathsf{proc}(x, \mathsf{send}\; x\; w \;;\; P) \quad \mathsf{proc}(z, y \leftarrow \mathsf{recv}\; x \;;\; Q_y)}{\mathsf{proc}(P) \quad \mathsf{proc}(Q_w)}\;\bullet C, \circ C
$$