Constructive Logic (15-317), Spring 2022
Assignment 11: Modal, Linear, and Ordered Logic

Instructor: Klaas Pruiksma
TAs: Runming Li, Onyekachi Onyeador, Viraj Puri, Xiao Yu

Due: Thursday, April 28, 11:59 pm

The assignments in this course must be submitted electronically through Gradescope. For this homework, you will be submitting both written pdf files and Dcheck coding files:

- `hw.deriv` (your coding solutions)
- `hw.pdf` (your written solutions)

The coding portion will use the experimental Dcheck derivation checker. You can find documentation and examples on the Software page at the course web site (`https://www.andrew.cmu.edu/user/kpruiksm/15317s22/dcheck.pdf`).

1 Fun with Modal Logic

Task 1 (20 pts)**.** Provide derivations for the following Modal Logic judgements using Dcheck syntax. Derivation names are given in the starter code.

1. $\Box T$ *true*
2. $\Box A \supset \Diamond\neg A \supset \Diamond F$ *true*
3. $(\Box A \wedge \Box B) \supset \Box(A \wedge B)$ *true*
4. $\Diamond(\Box A \wedge \Diamond(A \supset B)) \supset \Diamond B$ *true*
5. $\Diamond\Box\Diamond A \supset \Diamond A$ *true*

2 Practicing Linear Logic

Task 2 (16 pts)**.** Provide derivations for the following Linear Logic judgements using Dcheck syntax. Derivation names are given in the starter code.

1. $\Vdash (A \multimap B \multimap C) \multimap (A \otimes B \multimap C)$ *true*
2. $\Vdash (((A \otimes \top) \,\&\, (B \otimes \top)) \multimap C) \multimap (A \multimap B \multimap C)$ *true*
3. $\Vdash ((A \multimap C) \oplus (B \multimap C)) \multimap (A \,\&\, B) \multimap C$ *true*
4. $\Vdash ((A \multimap 0) \oplus (B \multimap 0)) \multimap (A \otimes B) \multimap 0$ *true*

3 Linear Harmony

Just like we did in the beginning of the course, we can check a local correctness condition for the rules of linear natural deduction: proof-theoretic harmony.[1] Hint: exhibiting local reductions and expansions in linear logic is subtle: you must be sure to not constrain the contexts Δ in your local reductions and expansions any more than is warranted by the rules of linear logic.

Remark 1 (Linear substitution principle)**.** When exhibiting local reductions and expansions, you will need to use substitutions $[\mathcal{D}/u]\mathcal{E}$. These are governed by the *linear substitution principle*, which states:

$$\text{If } \Gamma \Vdash \overset{\mathcal{D}}{A}\ \textit{true} \text{ and } \Gamma', u : A\ \textit{true} \Vdash \overset{\mathcal{E}}{B}\ \textit{true}, \text{ then } \Gamma, \Gamma' \Vdash \overset{[\mathcal{D}/u]\mathcal{E}}{B}\ \textit{true}.$$

You *must* ensure that your resulting derivations have correct contexts.

Task 3 (10 pts)**.** Verify that tensor $\otimes$ satisfies local soundness and completeness.

Task 4 (10 pts)**.** Verify that that **1** satisfies local soundness and completeness.

[1] Harmony is a necessary condition for the correctness of rules, but not a sufficient condition.

Task 5 (10 pts). Verify that $\top$ satisfies local soundness and completeness.

The relevant rules are:

$$\frac{\Delta \Vdash A\ \mathit{true} \quad \Delta' \Vdash B\ \mathit{true}}{\Delta, \Delta' \Vdash A \otimes B\ \mathit{true}} \otimes\mathsf{I} \qquad \frac{\Delta \Vdash A \otimes B\ \mathit{true} \quad \Delta', u : A\ \mathit{true}, v : B\ \mathit{true} \Vdash C\ \mathit{true}}{\Delta, \Delta' \Vdash C\ \mathit{true}} \otimes\mathsf{E}^{u,v}$$

$$\frac{}{\cdot \Vdash \mathbf{1}\ \mathit{true}} \mathbf{1}\mathsf{I} \qquad \frac{\Delta \Vdash \mathbf{1}\ \mathit{true} \quad \Delta' \Vdash C\ \mathit{true}}{\Delta, \Delta' \Vdash C\ \mathit{true}} \mathbf{1}\mathsf{E} \qquad \frac{}{\Delta \Vdash \top\ \mathit{true}} \top\mathsf{I}$$

4 Applications

In class, we looked at *Blocks World*, an example of encoding *state* in linear logic. Blocks World is a class of scenarios in which there is a table, some number of blocks which can be stacked on top of each other, and a robotic arm which can pick up and move blocks. The following atomic predicates are used:

1. empty means that the robotic arm's hand is empty.
2. $\mathsf{holds}(x)$ means that the robotic arm's hand is holding x.
3. $\mathsf{clear}(x)$ means that the block x does not have anything on top of it.
4. $\mathsf{on}(x, y)$ means that the block x is directly on top of the block y.
5. $\mathsf{on\_table}(x)$ means that the block x is sitting directly on the table.

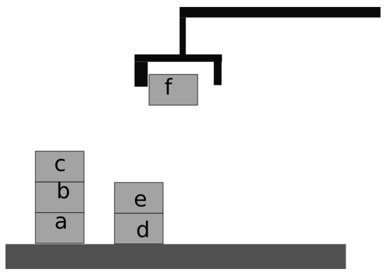
The possible state transitions for Blocks World are given by the following axioms:

1. $\forall x, y.\ \mathsf{empty} \otimes \mathsf{clear}(x) \otimes \mathsf{on}(x, y) \multimap \mathsf{holds}(x) \otimes \mathsf{clear}(y)$
2. $\forall x. \mathsf{empty} \otimes \mathsf{clear}(x) \otimes \mathsf{on\_table}(x) \multimap \mathsf{holds}(x)$
3. $\forall x, y.\ \mathsf{holds}(x) \otimes \mathsf{clear}(y) \multimap \mathsf{empty} \otimes \mathsf{on}(x, y) \otimes \mathsf{clear}(x)$
4. $\forall x.\ \mathsf{holds}(x) \multimap \mathsf{empty} \otimes \mathsf{on\_table}(x) \otimes \mathsf{clear}(x)$

Task 6 (10 pts). So far we have assumed that the table is infinitely broad and can therefore accomodate any number of blocks. **Now consider the case that the table in fact only has a finite number of spaces for blocks on it, and modify the axioms of Blocks World above in order to preserve this invariant.**

You should use an atomic predicate space, which represents one open horizontal space on the table. A correct solution to this task will not depend on the size of the table.

Task 7 (10 pts). Consider the following Blocks World scenario:



Write a formula in linear logic which expresses this configuration, assuming that **four** blocks can fit directly on the table.

5 Ordered Logic

In ordered natural deduction, we had the following elimination rule for the *under* connective:

$$\frac{A \quad A \setminus B}{B}\ \backslash\mathsf{E}$$

In the ordered sequent calculus, however, we had the following left rule:

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

We could also have tried another version of the left rule, which we will call $\backslash\mathsf{L}'$, which more closer mirrored the natural deduction elimination form:

$$\frac{\Omega_L\ B\ \Omega_R \Longrightarrow C}{\Omega_L\ A\ (A \setminus B)\ \Omega_R \Longrightarrow C}\ \backslash\mathsf{L}'$$

Task 8 (10 pts)**.** Assuming *cut*, show that these two versions of the left rule are equivalent. That is, show that $\backslash\mathsf{L}'$ is a derived rule for the sequent calculus which has just $\backslash\mathsf{L}$, and show that $\backslash\mathsf{L}$ is a derived rule for the sequent calculus which has just $\backslash\mathsf{L}'$.

Task 9 (10 pts)**.** As is the case with many seemingly sensible rules, the new left rule $\backslash\mathsf{L}'$ will actually destroy the logical character of the sequent calculus. In particular, the version of sequent calculus with $\backslash\mathsf{L}'$ and without $\backslash\mathsf{L}$ does not enjoy the admissibility of cut, and thence does not validate the cut elimination theorem.

Demonstrate a counterexample to cut elimination in the calculus with $\backslash\mathsf{L}'$; to be precise, this should be a sequent $\Gamma \Longrightarrow C$ for some specific Γ and specific C which can be proved with *cut* but has no cut-free proof.

6 Subsingleton Logic

Consider the encoding of binary numbers in ordered inference where the proposition `b0` represents a 0 bit, `b1` represents a 1 bit, and \$ represents the the end of a binary string; in this encoding, we represent the binary number 1101 with the ordered state \$ `b1 b0 b1 b1`.

Task 10 (10 pts)**.** Use *ordered inference rules* to encode a procedure that decides whether a binary string contains an even or odd number of 1-bits in it.

Specifically, introducing a new atomic proposition par together with rules such that when $\vec{A}$ is the encoding of a binary string in standard form, one can derive $\frac{\vec{A} \quad \mathsf{par}}{\$ \quad \mathsf{b0}}$ iff the string $\vec{A}$ has an even number of 1-bits, and one can derive $\frac{\vec{A} \quad \mathsf{par}}{\$ \quad \mathsf{b1}}$ iff $\vec{A}$ has an odd number of 1-bits.

You may freely introduce any auxiliary propositions and rules that you require.

Task 11 (10 pts)**.** Rewrite your solution to the previous task in the form of a well-typed ordered concurrent program in the subsingleton fragment. You may freely make any definitions you require, including recursive definitions of session types and concurrent programs.

Your solution should include the definition of a type of bit strings $bits$, together with a program par that has the type $bits \vdash \mathsf{par} : bits$.