Constructive Logic (15-317), Fall 2018
Assignment 9: Focusing and Classical Logic

Course Staff

Due: Friday, November 16, 11:59 pm

Submit your homework via autolab as a file named **hw9.pdf**.

Focusing and Chaining

A major theme of this course has been the discovery of theory through practice: strategies for efficient proof search in the concrete conditions of real-world implementations are transformed into razor-edged intellectual weapons, entirely new logics which sharpen the principal contradiction of proof theory: the dialectic of the *positive* and *negative* (polarity).

The decomposition of *truth* into *verification* and *use* was our first encounter with the scientific law, "One Divides Into Two". By studying invertibility in the context of the sequent calculus (when does a conclusion imply its premises?), we were able to achieve a firmer grasp of the fault-lines at play, summarized in a dangerously over-simplified[1] form below:

	LEFT RULE	RIGHT RULE
POSITIVE	invertible	**non-invertible**
NEGATIVE	**non-invertible**	invertible

Inversion Invertible rules can always be applied without any need for backtracking: since the conclusion of an invertible rule implies its premises, the "future truth" of the goal is preserved under free application of such rules. This practical insight, which is crucial for implementing a performant proof search engine, can be codified by sharpening the logic to include deterministic inversion phases $\Gamma; \Omega \longrightarrow_{\mathsf{L}} C$ and $\Gamma; \Omega \longrightarrow_{\mathsf{R}} C$ (where Ω is an ordered context of propositions).

Chaining While the above gives a clear and deterministic account of invertible rules, the non-invertible ones beg for something similar. In this week's lecture, we began to study *chaining*, which fixes a dynamics for the non-invertible rules based on two forms of judgment, $\Gamma \longrightarrow [A^+]$ and $\Gamma; [A^-] \longrightarrow C$. Chaining is a technique to minimize backtracking by applying a sequence of non-invertible rules in one go.

[1] In *structural* or *persistent* logic, some rules which ought to be non-invertible turn out to be invertible; polarity arises properly from the proof search dynamics of *linear logic*, and casts an imperfect shadow in persistent logic.

1 Practicing focusing

Task 1 (10 pts)**.** Construct a derivation in focused logic for the following sequent:

$$\cdot;\cdot \longrightarrow_{\mathsf{R}} {\downarrow}\big(\big(a^+ \supset b^-\big) \wedge \big(a^+ \supset c^-\big)\big) \supset \big(a^+ \supset \big(b^- \wedge c^-\big)\big)$$

Task 2 (20 pts)**.** Consider the following depolarized formula:

$$\neg(a^+ \vee b^-) \supset \neg a^+ \wedge \neg b^-$$

Come up with two *distinct* polarizations of the formula, adding shifts in the appropriate places; you do not need to prove them. Hint: remember that in depolarized constructive logic, negation $\neg A \equiv A \supset \bot$; in your solution, you must choose a polarization for negations.

2 Saturation

Consider the following grammar of ground terms representing binary numbers:

$$n \quad ::= \quad \epsilon \mid \mathtt{b0}(n) \mid \mathtt{b1}(n)$$

In class, we learned to write forward logic programs using inference rules; a forward logic programming engine will apply these inference rules until saturation is reached, and then the result of our program can be read from the saturated proof state. In the tasks that follow, you are free to introduce any auxiliary predicates that you require. You need to ensure that your rules *saturate* when new facts of the indicated form are added to the database.

In the problems that follow, you are required to implement forward logic programs by writing down systems of inference rules. You may find it useful to experiment with **DLV**, an implementation of forward logic programming which can be downloaded here: `http://www.dlvsystem.com/dlv/`. **DLV** can be used to test your ideas on specific cases and quickly determine if they are likely to work; but it is not required.

Task 3 (5 pts)**.** Implement a forward logic program $\mathsf{std}(n)$ which derives the atom no iff it is not the case that n is in standard form. You may assume that n is ground (i.e. not subject to unification).

Task 4 (5 pts)**.** Next, implement a forward logic program $\mathsf{succ}(m, n)$ which derives no when it is not the case that $m + 1 = n$. For the purpose of this exercise, you may assume that m and n are ground. You may also assume that m and n are in standard form.

3 A New Constructive Logic: Classical Logic

Intuitionistic logic is based on the idea that the fundamental mathematical activity is to *affirm* the truth of something using evidence. Classical logic should be understood as a different, *dialectical* model of mathematical activity, in which one party tries to affirm and the other party tries to deny. Whereas the central duality of intuitionistic natural deduction was between the *introduction* and *elimination* rules for truth $A\ \mathit{true}$, in classical natural deduction, each proposition is explained through the interaction between rules for affirmation $A\ \text{👍}$ and rules for denial $A\ \text{👎}$, a contest governed by the nullary form of judgment $\#$ (contradiction).[2]

To be precise, each connective comes equipped with introduction rules for *both* affirmation and denial; classical negation $\neg A$ implements the involutive "change of perspective" between player (affirmation) and opponent (denial).

$\boxed{\text{Conjunction}}$

$$\frac{A\ \text{👍} \quad B\ \text{👍}}{A \wedge B\ \text{👍}}\ \wedge\text{👍} \qquad \frac{A\ \text{👎}}{A \wedge B\ \text{👎}}\ \wedge\text{👎}_1 \qquad \frac{B\ \text{👎}}{A \wedge B\ \text{👎}}\ \wedge\text{👎}_2$$

$\boxed{\text{Disjunction}}$

$$\frac{A\ \text{👍}}{A \vee B\ \text{👍}}\ \vee\text{👍}_1 \qquad \frac{B\ \text{👍}}{A \vee B\ \text{👍}}\ \vee\text{👍}_2 \qquad \frac{A\ \text{👎} \quad B\ \text{👎}}{A \vee B\ \text{👎}}\ \vee\text{👎}$$

$\boxed{\text{Implication}}$

$$\frac{\begin{array}{c} \overline{A\ \text{👍}}\ u \\ \vdots \\ B\ \text{👍} \end{array}}{A \supset B\ \text{👍}}\ \supset\text{👍}^u \qquad \frac{A\ \text{👍} \quad B\ \text{👎}}{A \supset B\ \text{👎}}\ \supset\text{👎}$$

$\boxed{\text{Units}}$

$$\frac{}{\top\ \text{👍}}\ \top\text{👍} \qquad \frac{}{\bot\ \text{👎}}\ \bot\text{👎}$$

$\boxed{\text{Negation}}$

$$\frac{A\ \text{👎}}{\neg A\ \text{👍}}\ \neg\text{👍} \qquad \frac{A\ \text{👍}}{\neg A\ \text{👎}}\ \neg\text{👎}$$

In classical natural deduction, affirmation and denial compete with each other in a *formal contradiction*, a nullary judgment written $\boxed{\#}$. The rules for contradictions are as follows:

[2]The symbols $A\ \text{👍}$ and $A\ \text{👎}$ can be written using the provided macros; if you have trouble using these symbols, it is also acceptable to write A yes and A no.

$\boxed{\text{Contradiction}}$

$$
\dfrac{A\,\text{👍} \qquad A\,\text{👎}}{\#}\ \text{link}
\qquad\qquad
\dfrac{\begin{array}{c}\overline{A\,\text{👍}}^{\,u}\\ \vdots \\ \#\end{array}}{A\,\text{👎}}\ \#\text{👎}^{u}
\qquad\qquad
\dfrac{\begin{array}{c}\overline{A\,\text{👎}}^{\,u}\\ \vdots \\ \#\end{array}}{A\,\text{👍}}\ \#\text{👍}^{u}
$$

Using the rules $\#$👍 and $\#$👎, all the usual "elimination rules" for truth can be *derived* in classical natural deduction.

Task 5 (10 pts)**.** Recall the introduction and elimination rules for the universal quantifier in intuitionistic natural deduction:

$$
\dfrac{\begin{array}{c}[z:\tau]\\ \vdots \\ A(z)\ \mathit{true}\end{array}}{\forall x:\tau.\ A(x)\ \mathit{true}}\ \forall\mathsf{I}^{z}
\qquad\qquad
\dfrac{t:\tau \qquad \forall x:\tau.\ A(x)\ \mathit{true}}{A(t)\ \mathit{true}}\ \forall\mathsf{E}
$$

Now it's your turn: *invent* affirmation and denial rules $\forall$👍, $\forall$👎 for the universal quantifier, as an extension to the classical natural deduction calculus which we have seen so far.

Task 6 (10 pts)**.** Recall the introduction and elimination rules for the existential quantifier in intuitionistic natural deduction:

$$
\dfrac{t:\tau \qquad A(t)\ \mathit{true}}{\exists x:\tau.\ A(x)\ \mathit{true}}\ \exists\mathsf{I}
\qquad\qquad
\dfrac{\exists x:\tau.\ A(x)\ \mathit{true} \qquad \begin{array}{c}[z:\tau]\quad \overline{A(z)\ \mathit{true}}^{\,u}\\ \vdots \\ C\ \mathit{true}\end{array}}{C\ \mathit{true}}\ \exists\mathsf{E}^{z,u}
$$

As in the previous task, invent affirmation and denial rules $\exists$👍, $\exists$👎 for the existential quantifier.

Task 7 (10 pts)**.** Using the rules you invented in the previous tasks, show that the following *elimination* rules for the universal and the extistential quantifier are derivable.

$$
\dfrac{t:\tau \qquad \forall x:\tau.\ C(x)\,\text{👍}}{C(t)\,\text{👍}}\ \forall\text{👍}\mathsf{E}
\qquad\qquad
\dfrac{\exists x:\tau.\ A(x)\,\text{👍} \qquad \begin{array}{c}[z:\tau]\quad \overline{A(z)\,\text{👍}}^{\,u}\\ \vdots \\ C\,\text{👍}\end{array}}{C\,\text{👍}}\ \exists\text{👍}\mathsf{E}^{z,u}
$$