Constructive Logic (15-317), Fall 2015
Assignment 7: Sequent Calculus for Proof Search

Michael Coblenz (mcoblenz@cs.cmu.edu);
with credit to Joe Tassarotti and Evan Cavallo

Out: Tuesday, October 20, 2015
Due: Tuesday, October 27, 2015 (before class)

In this assignment, you will explore the **G4ip** sequent calculus and see how it may be used to build a simple yet realistic theorem prover for intuitionistic propositional logic. By the end of the assignment, you will have implemented a sound and complete proof search procedure capable of proving automatically any of the propositional theorems you've proven manually this semester using Tutch.

The written and programming portion of your work (Section 3) should be submitted via AFS by putting your PDF and code in the directory

`/afs/andrew/course/15/317/submit/<userid>/hw07`

where `<userid>` is replaced with your Andrew ID.

1 Invertibility (6 points)

Consider the two connectives $\heartsuit(A, B, C)$ and $\diamond(A, B, C)$ defined below:

$$\frac{\begin{array}{cc}[A\ true] & [B\ true] \\ \vdots & \vdots \\ C\ true & C\ true\end{array}}{\heartsuit(A, B, C)\ true}\ \heartsuit I \qquad \frac{\heartsuit(A, B, C)\ true \quad A\ true}{C\ true}\ \heartsuit E_1 \qquad \frac{\heartsuit(A, B, C)\ true \quad B\ true}{C\ true}\ \heartsuit E_2$$

$$\frac{\begin{array}{c}[A\ true] \\ \vdots \\ C\ true\end{array}}{\diamond(A, B, C)\ true}\ \diamond I_1 \qquad \frac{\begin{array}{c}[B\ true] \\ \vdots \\ C\ true\end{array}}{\diamond(A, B, C)\ true}\ \diamond I_2 \qquad \frac{\diamond(A, B, C)\ true \quad A\ true \quad B\ true}{C\ true}\ \diamond E$$

Task 1 (6 pts)**.** For each of the above rules, say whether it is invertible or not. Explain your answers.

2 Manual Theorem Proving (10 points)

In order to get some practice using the G4ip system, write proofs for these propositions using the rules of G4ip (see Appendix A). Assume that P, Q, R, and S stand for atomic propositions.

Task 2. (5 points)

$$\longrightarrow ((P \supset Q) \supset R) \wedge ((P \supset Q) \supset S) \supset (P \supset Q) \supset R$$

Task 3. (5 points)

$$\longrightarrow (((P \vee Q) \wedge R) \supset S) \supset R \wedge (Q \vee P) \supset S$$

3 Automated Theorem Proving (24 points)

Because **G4ip**'s rules all reduce the "weight" of the formulas making up the sequent when read bottom-up, it is straightforward to see that it represents a decision procedure even without the benefit of loop checking. The rules themselves are non-deterministic, though, so one must invest some effort in extracting a deterministic implementation from them.

Task 4 (25 pts)**.** Implement a proof search procedure based on the **G4ip** calculus. Efficiency should not be a primary concern, but see the hints below regarding invertible rules. Strive instead for *correctness* and *elegance*, in that order.

You should write your implementation in Standard ML.[1] Some starter code is provided in the file `prop.sml` to clarify the setup of the problem and give you some basic tools for debugging (see Figure 1). Implement a structure `G4ip` matching the signature `G4IP`. A simple test harness assuming this structure is given in the structure `Test` in the file `test.sml`. Feel free to post any additional interesting test cases you encounter to the course bulletin board.

Here are some hints to help guide your implementation:

- Be sure to apply all invertible rules before you apply any non-invertible rules. Recall that the only non-invertible rules in **G4ip** are $\vee R_1$, $\vee R_2$, and $\supset\supset L$, but that $P\supset L$ and the init rule cannot always be applied asynchronously. One simple way to ensure that you do inversions first is to

[1] If you are not comfortable writing in Standard ML, you should contact the instructors and the TA to work out an alternate arrangement.

```
signature PROP =
  sig
    datatype prop =                 (* A ::=            *)
        Atom of string              (*       P          *)
      | True                        (*     | T          *)
      | And of prop * prop          (*     | A1 & A2    *)
      | False                       (*     | F          *)
      | Or of prop * prop           (*     | A1 | A2    *)
      | Implies of prop * prop      (*     | A1 => A2   *)

    val Not : prop -> prop          (* ~A := A => F     *)

    val toString : prop -> string
  end

structure Prop :> PROP = ...

signature G4IP =
  sig
    (* [decide A = true] iff . ===> A has a proof,
       [decide A = false] iff . ===> A has no proof *)
    val decide : Prop.prop -> bool
  end
```

Figure 1: SML starter code for **G4ip** theorem prover.

maintain a second context of non-invertible propositions and to process it only when the invertible context is exhausted.

- When it comes time to perform non-invertible search, you'll have to consider all possible choices you might make. Many theorems require you to use your non-invertible hypotheses in a particular order, and unless you try all possible orders, you may miss a proof.
- The provided test cases can help you catch many easy-to-make errors. Test your code early and often! If you come up with any interesting test cases of your own that help you catch other errors, we encourage you to share them via the course bulletin board.

There are many subtleties and design decisions involved in this task, so don't leave it until the last minute!

A Complete G4ip Rules

Init Rule

$$\frac{}{\Delta, P \longrightarrow P}\ \text{init}$$

Ordinary Rules

$$\frac{}{\Delta \longrightarrow \top}\ \top R \qquad \frac{\Delta \longrightarrow C}{\Delta, \top \longrightarrow C}\ \top L$$

$$\frac{\Delta \longrightarrow A \quad \Delta \longrightarrow B}{\Delta \longrightarrow A \wedge B}\ \wedge R \qquad \frac{\Delta, A, B \longrightarrow C}{\Delta, A \wedge B \longrightarrow C}\ \wedge L$$

(no $\bot R$ rule) $\qquad \dfrac{}{\Delta, \bot \longrightarrow C}\ \bot L$

$$\frac{\Delta \longrightarrow A}{\Delta \longrightarrow A \vee B}\ \vee R_1 \quad \frac{\Delta \longrightarrow B}{\Delta \longrightarrow A \vee B}\ \vee R_2 \qquad \frac{\Delta, A \longrightarrow C \quad \Delta, B \longrightarrow C}{\Delta, A \vee B \longrightarrow C}\ \vee L$$

$$\frac{\Delta, A \longrightarrow B}{\Delta \longrightarrow A \supset B}\ \supset R$$

Compound Left Rules

$$\frac{P \in \Delta \quad \Delta, B \longrightarrow C}{\Delta, P \supset B \longrightarrow C}\ P \supset L$$

$$\frac{\Delta, B \longrightarrow C}{\Delta, \top \supset B \longrightarrow C}\ \top \supset L \qquad \frac{\Delta, D \supset E \supset B \longrightarrow C}{\Delta, D \wedge E \supset B \longrightarrow C}\ \wedge \supset L$$

$$\frac{\Delta \longrightarrow C}{\Delta, \bot \supset B \longrightarrow C}\ \bot \supset L \qquad \frac{\Delta, D \supset B, E \supset B \longrightarrow C}{\Delta, D \vee E \supset B \longrightarrow C}\ \vee \supset L$$

$$\frac{\Delta, D, E \supset B \longrightarrow E \quad \Delta, B \longrightarrow C}{\Delta, (D \supset E) \supset B \longrightarrow C}\ \supset \supset L$$