Constructive Logic (15-317), Fall 2018
Assignment 7: Proof search in G4ip

Course Staff

Due: Friday, October 26, 2018, 11:59 pm

Submit your homework as a single file, **not a tar**. Just submit `g4ip.sml`. **After submitting via autolab, please check the submission's contents to ensure it contains what you expect. No points can be given to a submission that isn't there.**

Last week, we implemented inversion calculus for the implication-free fragment of propositional logic. However, implications are important! Hence, this week we will be implementing Roy Dyckhoff's contraction-free sequent calculus. This calculus, called **g4ip**, relies on distinguishing the type of antecedent on an implication on the left. To perform efficient proof search, we are extending the inversion calculus with a new form of judgment to apply synchronous (non-invertible) rules; these include the right rules of right-synchronous connectives and the left rules of left-synchronous connectives. For reference, the rules are below. For further information, please see the recitation notes for this week. They have an excellent description for the rules of **g4ip**, along with directions for implementing them as a decision procedure in a functional programming language.

Our forms of judgment are as follows:

$$\begin{array}{ll}
\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} C & \text{Decompose } C \text{ on the right} \\
\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+ & \text{Decompose } \Omega \text{ on the left} \\
\Gamma^- \xrightarrow{\mathsf{g4ip}} C^+ & \text{Apply non-invertible rules}
\end{array}$$

The rules of our version of **g4ip** are divided roughly into the inversion phases (right and left) and the search phase.

Right Inversion

$$\frac{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} A \quad \Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} B}{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} A \wedge B} \wedge\mathsf{R} \qquad \frac{\Gamma^-; \Omega, A \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} B}{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} A \supset B} \supset\mathsf{R} \qquad \frac{}{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} \top} \top\mathsf{R}$$

Switching Mode

$$\frac{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-; \Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} C^+} \mathsf{LR}_+$$

Left Inversion

$$\frac{\Gamma^-; \Omega, A, B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-; \Omega, A \wedge B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+} \wedge\mathsf{L} \qquad \frac{\Gamma^-; \Omega, A \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+ \quad \Gamma^-; \Omega, B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-; \Omega, A \vee B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+} \vee\mathsf{L} \qquad \frac{}{\Gamma^-; \Omega, \bot \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+} \bot\mathsf{L}$$

$$\frac{\Gamma^-;\Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,\top \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \top\mathsf{L}$$

Compound Left Invertible Rules

$$\frac{\Gamma^-;\Omega,B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,\top\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \top{\supset}\mathsf{L} \qquad \frac{\Gamma^-;\Omega,A_1\supset A_2\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,(A_1\wedge A_2)\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \wedge{\supset}\mathsf{L}$$

$$\frac{\Gamma^-;\Omega,A_1\supset B,A_2\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,(A_1\vee A_2)\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \vee{\supset}\mathsf{L} \qquad \frac{\Gamma^-;\Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,\bot\supset B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \bot{\supset}\mathsf{L}$$

Shift and Search

$$\frac{\Gamma^-,A^-;\Omega \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-;\Omega,A^- \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \mathsf{shift} \qquad \frac{\Gamma^- \xrightarrow{\mathsf{g4ip}} C^+}{\Gamma^-;\cdot \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}\ \mathsf{search}$$

Search Rules

$$\frac{P\in\Gamma^-}{\Gamma^- \xrightarrow{\mathsf{g4ip}} P}\ \mathsf{init} \qquad \frac{\Gamma^-;\cdot \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} A}{\Gamma^- \xrightarrow{\mathsf{g4ip}} A\vee B}\ \vee\mathsf{R}_1 \qquad \frac{\Gamma^-;\cdot \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} B}{\Gamma^- \xrightarrow{\mathsf{g4ip}} A\vee B}\ \vee\mathsf{R}_2$$

Compound Left Search Rules

$$\frac{P\in\Gamma^- \quad \Gamma^-;B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-,P\supset B \xrightarrow{\mathsf{g4ip}} C^+}\ P{\supset}\mathsf{L} \qquad \frac{\Gamma^-;A_2\supset B,A_1 \xrightarrow{\mathsf{g4ip}}_{\mathsf{R}} A_2 \quad \Gamma^-;B \xrightarrow{\mathsf{g4ip}}_{\mathsf{L}} C^+}{\Gamma^-,(A_1\supset A_2)\supset B \xrightarrow{\mathsf{g4ip}} C^+}\ {\supset}{\supset}\mathsf{L}$$

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.

Tip The rules themselves are non-deterministic, so one must invest some effort in extracting a deterministic implementation from them.

Task 1. Implement a proof search procedure based on **g4ip**. 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`, included in this homework's handout, to clarify the setup of the problem and give you some basic tools for debugging.

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

---

[1] If you are not comfortable writing in Standard ML, you should contact the instructors and the TAs for help.

```
    | False                     (*     | F          *)
    | Or of prop * prop         (*     | A1 | A2    *)
    | Imp of prop * prop        (*     | A1 => A2   *)

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

  val toString : prop -> string
  val toStringList : prop list -> string
  val eq : prop * prop -> bool
end

structure Prop :> PROP
```

Your task is implement a structure `G4ip` matching the signature `G4IP`. The signature has been provided below, and is included in the handout materials.

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

A simple test harness assuming this structure is given in the structure `Test` in the file `test.sml`, also included in the handout. Feel free to post any additional interesting test cases you encounter to Piazza.

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 non-invertible rules in **g4ip** are init, $\vee R_1$, $\vee R_2$, $P{\supset}L$ and $\supset{\supset}L$. Among these, init and $P{\supset}L$ have somewhat special status: if they apply, we don't need to look back because there is no premise (init), or the sequent in the premise is provable whenever the conclusion is ($P{\supset}L$).

  One simple way to ensure that you do inversions first is to maintain a second context of non-invertible propositions and to process it only when the invertible rules have been 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 on Piazza.

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