AMSI
VACATION
RESEARCH
SCHOLARSHIPS
2018-2019

Equivalence of Natural Deduction and Sequent Calculus in HOL4

Alexander Cox
The Australian National University (ANU)

Supervised by Dr Michael Norrish
Data61, CSIRO; ANU

February 28, 2019

Vacation Research Scholarships are funded jointly by the Department of Education and Training and the Australian Mathematical Sciences Institute.

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

Australian Government
Department of Education and Training

Abstract

I describe the mechanisation of the equivalence of two proof calculi, natural deduction and sequent calculus, for intuitionistic propositional logic using the HOL4 interactive theorem prover. The equivalence of these calculi shows that given the same hypotheses, the same conclusion can be deduced by both calculi. This result is achieved by rule induction on the inference rules of each system. I present the relevant proof theory background and its formalisation in HOL4.

1 Introduction

In this project I have closely followed sections of Troelstra and Schwichtenberg (2000) in its presentation of Natural Deduction (**N**) and Sequent Calculus (**G**). I will note any deviance from this book.

N and **G** are logical calculi introduced by Gentzen in the mid 1930s. **N** derives formulae from assumptions using introduction or elimination rules (which either introduce or eliminate a logical operator). **G** derives sequents, which are a multisets of formulae related by the sequent relation (denoted $\vdash$). **G** has one axiom, in addition to left and right rules which operate on formulae in either the left or right of the sequent. I have been formalising these proof systems for intuitionistic propositional logic. In particular I have been mechanising the proof of equivalence between these two calculi.

The formalisation has taken place in the HOL4 Theorem Prover (Slind and Norrish 2008), henceforth referred to as HOL.

The purpose of this project was for me to learn how to use an interactive theorem prover. The proof theory itself is well known, and has been formalised before, albeit in a slightly different manner.

1.1 The HOL Theorem Prover

HOL is an interactive theorem prover which implements Higher Order Logic as the meta-logic with which users formalise mathematics. HOL implements Church's Simple Theory of Types with polymorphic types (ibid.). HOL is implemented with Standard ML, and this is the meta-language and main interface to HOL.

Proving a theorem in HOL guarantees that your proof is correct and the theorem is sound, under the assumption that HOL is itself sound. All theorems in HOL are generated from the composition of a small set of axioms and basic inference rules, which are considered trusted by the HOL community. HOL can be built using two different Standard ML compilers, which increases confidence that the soundness of HOL is compiler-independent. HOL can also produce a certificate of correctness which can then be checked by another theorem prover, as an additional measure to establish trust in soundness. Saying this, any proof relies on first having a correct specification and formulation of the mathematical content. If I have not formalised my definitions correctly, my proofs don't establish the truth of theorems concerning the correct definitions.

HOL has good support for inductively defined relations (Camilleri and Melham 1992), which I use in this project. HOL automatically proves a strong induction theorem for the defined relations, which can then be used to perform rule induction on the relation (ibid.).

1.2 Related Work

The equivalence of Natural Deduction, Sequent Calculus and Hilbert calculus for classical propositional logic, has been formalised in the theorem prover Coq, by Doorn (2015). A major difference between my formalisation and that of Doorn is that they used lists for their contexts in both **N** and **G**,

whereas I have used sets and multisets respectively. They also mechanised the proofs of soundness and completeness.

The same equivalence, but for first order classical logic, has been formalised in HOL by Mikhajlova and Wright (1998). They also mechanised the proofs of deduction monotonicity and compactness, amongst others. The latter was formalised quite differently to the way I have formalised the calculi in this project, using several additional purpose-built datatypes, such as one for derivations.

2 Formalisation in HOL

2.1 Syntax

I formalised the intuitionistic logic versions of **N** and **G**, referred to as **Ni** and **G2i** in Troelstra and Schwichtenberg (2000).

Definition 2.1.1 (Formula)**.** The formulae of intuitionistic propositional logic are defined inductively, starting with atomic variables of arbitrary type α, then connected with logical operators. I have used non-standard symbols $\veebar$ and $\barwedge$ in place of $\vee$ and $\wedge$ to avoid confusion with the meta-logical symbols used. The operators are defined in prefix form, but are always used as infix operators afterwards[1].

Here is how formulae are defined in BNF syntax:

$$\varphi ::= \alpha \mid \varphi \veebar \varphi \mid \varphi \barwedge \varphi \mid \varphi \to \varphi \mid \bot$$

where φ is a formula and α is an atomic variable.

Here is the corresponding HOL definition:

$$\begin{aligned}
&\alpha\,\texttt{formula} = \\
&\quad \mathsf{Var}\,\alpha \\
&\mid (\veebar)\,(\alpha\,\texttt{formula})\,(\alpha\,\texttt{formula}) \\
&\mid (\barwedge)\,(\alpha\,\texttt{formula})\,(\alpha\,\texttt{formula}) \\
&\mid (\to)\,(\alpha\,\texttt{formula})\,(\alpha\,\texttt{formula}) \\
&\mid \bot
\end{aligned}$$

Notation 2.1.2. The latter definition is produced by HOL for typesetting. For illustration purposes, here is how I wrote it in my text editor:

```
val _ = Datatype `formula =
    Var 'a
    | Or formula formula
    | And formula formula
    | Imp formula formula
    | Bot`;
```

For the remainder of the report I will use typeset versions of definitions and theorems produced by HOL.

Notation 2.1.3. I use A, B, C for arbitrary formulae.

[1] This is how they are defined in HOL

Definition 2.1.4 (Abbreviations)**.** The remainder of propositional logic syntax is achieved with abbreviations:

$$\vdash \forall A.\, \neg A = A \to \bot$$
$$\vdash \forall A\, B.\, A \leftrightarrow B = (A \to B) \barwedge (B \to A)$$
$$\vdash \top = \bot \to \bot$$

2.2 Natural Deduction (N)

I have represented **N** in sequent style (Troelstra and Schwichtenberg 2000, s. 2.1.8) using the complete discharge convention (ibid., s. 2.1.9). This means that rather than having a tree with assumptions as leaves, I have sequents, with a set of open assumptions on the left. Open assumptions are assumptions which have not been discharged.

The complete discharge convention says that I can discharge all instances of a assumption at once, rather than keeping track of assumptions with labels. This simplifies the presentation of Natural Deduction. The rules which discharge assumptions are $\to$i and $\veebar$e, which I define below.

Notation 2.2.1. $\Gamma \vdash_S A$ denotes that the formula A can be derived from the hypotheses Γ in proof system S. A $\vdash$ by itself denotes a theorem in the meta-logic, HOL.

Definition 2.2.2 (The **N** calculus)**.**

$$\frac{}{\{A\} \vdash_{\mathbf{N}} A}\ \mathsf{ax} \qquad \frac{D_1 \vdash_{\mathbf{N}} A \quad D_2 \vdash_{\mathbf{N}} B}{D_1 \cup D_2 \vdash_{\mathbf{N}} A \barwedge B}\ \barwedge\mathsf{i} \qquad \frac{D \vdash_{\mathbf{N}} A \barwedge B}{D \vdash_{\mathbf{N}} A}\ \barwedge\mathsf{el} \qquad \frac{D \vdash_{\mathbf{N}} A \barwedge B}{D \vdash_{\mathbf{N}} B}\ \barwedge\mathsf{er}$$

$$\frac{\{A\} \cup D \vdash_{\mathbf{N}} B}{D \vdash_{\mathbf{N}} A \to B}\ \to\mathsf{i} \qquad \frac{D_1 \vdash_{\mathbf{N}} A \to B \quad D_2 \vdash_{\mathbf{N}} A}{D_1 \cup D_2 \vdash_{\mathbf{N}} B}\ \to\mathsf{e} \qquad \frac{D \vdash_{\mathbf{N}} \bot}{D \vdash_{\mathbf{N}} A}\ \bot\mathsf{e}$$

$$\frac{D \vdash_{\mathbf{N}} A}{D \vdash_{\mathbf{N}} A \veebar B}\ \veebar\mathsf{il} \qquad \frac{D \vdash_{\mathbf{N}} B}{D \vdash_{\mathbf{N}} A \veebar B}\ \veebar\mathsf{ir} \qquad \frac{D \vdash_{\mathbf{N}} A \veebar B \quad \{A\} \cup D_1 \vdash_{\mathbf{N}} C \quad \{B\} \cup D_2 \vdash_{\mathbf{N}} C}{D \cup D_1 \cup D_2 \vdash_{\mathbf{N}} C}\ \veebar\mathsf{e}$$

Notation 2.2.3. Equations in this text starting with a $\vdash$ are exported theorems from HOL, and have been specialised (universal quantifiers have been stripped). For example, the finiteness property for **N** is as follows, first specialised, then not-specialised:

Corollary 2.2.4 (**N** hypotheses are finite)*.*

$$\vdash D \vdash_{\mathbf{N}} A \Rightarrow \mathsf{finite}\, D$$
$$\vdash \forall D\, A.\, D \vdash_{\mathbf{N}} A \Rightarrow \mathsf{finite}\, D$$

I will use the specialised versions for the remainder of the text.

Definition 2.2.5 (The **Nd** calculus)**.** The definition of **N** in Troelstra and Schwichtenberg (ibid.) has different rules when discharging assumptions, which I formalised as **Nd**. Rather than having singleton unions above the line, they have singleton set differences below the line. Here are the rules which differ:

$$\frac{D \vdash_{\mathbf{Nd}} B}{D \setminus \{A\} \vdash_{\mathbf{Nd}} A \to B}\ \to\mathsf{i} \qquad \frac{D \vdash_{\mathbf{Nd}} A \veebar B \quad D_1 \vdash_{\mathbf{Nd}} C \quad D_2 \vdash_{\mathbf{Nd}} C}{D \cup (D_1 \setminus \{A\}) \cup (D_2 \setminus \{B\}) \vdash_{\mathbf{Nd}} C}\ \veebar\mathsf{e}$$

Lemma 2.2.6 (**N** weakening)**.** $\vdash D \vdash_{\mathbf{N}} A \Rightarrow \forall B.\, \{B\} \cup D \vdash_{\mathbf{N}} A$

Lemma 2.2.7 (**Nd** weakening)**.** $\vdash D \vdash_{\mathbf{Nd}} A \Rightarrow \forall B.\, \{B\} \cup D \vdash_{\mathbf{Nd}} A$

Proof. Both proofs are the same, just replace **N** for **Nd** for the other. The proof is by construction:

$$\dfrac{\dfrac{\vdots}{D \vdash_{\mathbf{N}} A} \quad \dfrac{}{\{B\} \vdash_{\mathbf{N}} B}\,\text{ax}}{\dfrac{\{B\} \cup D \vdash_{\mathbf{N}} A \barwedge B}{\{B\} \cup D \vdash_{\mathbf{N}} A}\,\barwedge\text{e}}\,\barwedge\text{i}$$

□

Weakening can be extended as much as you like:

Lemma 2.2.8 (**N** superset weakening)**.** $\vdash \mathsf{finite}\, D' \Rightarrow \forall\, D\, A.\, D \vdash_{\mathbf{N}} A \wedge D \subseteq D' \Rightarrow D' \vdash_{\mathbf{N}} A$

Lemma 2.2.9 (**Nd** superset weakening)**.** $\vdash \mathsf{finite}\, D' \Rightarrow \forall\, D\, A.\, D \vdash_{\mathbf{N}} A \wedge D \subseteq D' \Rightarrow D' \vdash_{\mathbf{N}} A$

Proof. By induction on the cardinality of D'. You can insert as many formulae as you like. □

Theorem 2.2.10 (**N** is equivalent to **Nd**)**.** *Given the same hypotheses, the same formulae can be derived from both formulations of natural deduction:*

$$\vdash D \vdash_{\mathbf{N}} A \iff D \vdash_{\mathbf{Nd}} A$$

Proof. (if) Proof by rule induction (see Camilleri and Melham 1992, pp. 6–7) on **N**. Automatic rewrites and first-order automated reasoning prove the cases which coincide. The $\to$i and $\veebar$e cases are proved by using the corresponding inference rule, and using **Nd** weakening.

The $\to$i case illustrates the construction from **N** to **Nd**, which is similar for $\veebar$e.

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\{A\} \cup D \vdash_{\mathbf{Nd}} B}\,(\text{IH})}{\{A\} \cup D \setminus \{A\} \vdash_{\mathbf{Nd}} A \to B}\,\to\text{i}}{D \setminus \{A\} \vdash_{\mathbf{Nd}} A \to B}\,(\text{set difference definition})}{D \vdash_{\mathbf{Nd}} A \to B}\,(\textbf{Nd}\text{ superset weakening})$$

(only if) Proof by rule induction on **Nd**. Automatic rewrites and first-order automated reasoning prove the cases which coincide. The $\to$i and $\veebar$e cases are proved using **N** weakening, then the corresponding inference rule.

Here is the construction for $\veebar$e, the $\to$i case is similar.

$$\dfrac{\dfrac{\vdots}{D \vdash_{\mathbf{N}} A \veebar B}\,(\text{IH}) \quad \dfrac{\dfrac{\vdots}{D_1 \vdash_{\mathbf{N}} C}\,(\text{IH})}{\{A\} \cup (D \setminus \{A\}) \vdash_{\mathbf{N}} C}\,(\textbf{N}\text{ superset wkn}) \quad \dfrac{\dfrac{\vdots}{D_2 \vdash_{\mathbf{N}} C}\,(\text{IH})}{\{B\} \cup (D \setminus \{B\}) \vdash_{\mathbf{N}} C}\,(\textbf{N}\text{ superset wkn})}{D \cup (D_1 \setminus \{A\}) \cup (D_2 \setminus \{B\}) \vdash_{\mathbf{N}} C}\,\veebar\text{e}$$

□

2.3 Sequent Calculus (G)

I am using **G2i** in this project, as that is what was used in the book for the equivalence proof (Troelstra and Schwichtenberg 2000, s. 3.1.6). **G2i** has the weakening rules absorbed into the axiom and absurdity rules, but contains distinct contraction rules. Unlike **G2c** (classical logic), the conclusion is a single formula, rather than a bag of formulae.

Definition 2.3.1. Bags (a.k.a. multisets) are sets with duplicates permitted. In HOL bag is a function type: bag:$\alpha \mapsto$ num, where α is a type variable.

Notation 2.3.2. The empty bag is denoted $\{\|\}$.

Remark 2.3.3. I use (Troelstra and Schwichtenberg 2000, lemma 3.1.8) to eliminate empty succedents which are possible in **G2i**, but cause added complexity in formalisation. The only conclusion this removes from the calculus is the empty bag. If I had not done this, the consequent of the $\bot$ rule would instead be $\{\|\}$, and the conclusions of the other rules would be singleton bags of formulae rather than formulae.

Notation 2.3.4. Elements of bags are separated by semicolons. For example, $\{\|A;\ B;\ B\|\}$ is the bag containing three elements, one occurrence of A and two of B.

Definition 2.3.5. The union of two bags, denoted $b \uplus c$ is the sum of the element counts.

$$\vdash\ b \uplus c\ =\ (\lambda x.\, b\, x\ +\ c\, x)$$

Definition 2.3.6 (The **G** Calculus)**.**

$$\frac{A \in \Gamma \quad \mathsf{finite}\,\Gamma}{\Gamma \vdash_{\mathbf{G}} A}\ \mathsf{ax} \qquad \frac{\bot \in \Gamma \quad \mathsf{finite}\,\Gamma}{\Gamma \vdash_{\mathbf{G}} A}\ \mathsf{L}\bot \qquad \frac{\{\|A;\ A\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}{\{\|A\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}\ \mathsf{cont}$$

$$\frac{\{\|A\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}{\{\|A \barwedge B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}\ \mathsf{L}\barwedge\mathsf{L} \qquad \frac{\{\|B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}{\{\|A \barwedge B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}\ \mathsf{L}\barwedge\mathsf{R} \qquad \frac{\Gamma \vdash_{\mathbf{G}} A \quad \Gamma \vdash_{\mathbf{G}} B}{\Gamma \vdash_{\mathbf{G}} A \barwedge B}\ \mathsf{R}\barwedge$$

$$\frac{\{\|A\|\} \uplus \Gamma \vdash_{\mathbf{G}} C \quad \{\|B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}{\{\|A \veebar B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}\ \mathsf{L}\veebar \qquad \frac{\Gamma \vdash_{\mathbf{G}} A}{\Gamma \vdash_{\mathbf{G}} A \veebar B}\ \mathsf{R}\veebar\mathsf{L} \qquad \frac{\Gamma \vdash_{\mathbf{G}} B}{\Gamma \vdash_{\mathbf{G}} A \veebar B}\ \mathsf{R}\veebar\mathsf{R}$$

$$\frac{\Gamma \vdash_{\mathbf{G}} A \quad \{\|B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}{\{\|A \to B\|\} \uplus \Gamma \vdash_{\mathbf{G}} C}\ \mathsf{L}\to \qquad \frac{\{\|A\|\} \uplus \Gamma \vdash_{\mathbf{G}} B}{\Gamma \vdash_{\mathbf{G}} A \to B}\ \mathsf{R}\to \qquad \frac{\Gamma \vdash_{\mathbf{G}} A \quad \{\|A\|\} \uplus \Delta \vdash_{\mathbf{G}} B}{\Gamma \uplus \Delta \vdash_{\mathbf{G}} B}\ \mathsf{cut}$$

Corollary 2.3.7 (Hypotheses in **G** are finite)**.** $\vdash \Gamma \vdash_{\mathbf{G}} A \Rightarrow \mathsf{finite}\,\Gamma$

2.4 Bag lemmata

Recall that bags are multisets, defined as a characteristic function returning the number of occurrences of a given element. This definition comes from the provided 'bag theory' in HOL, which defines and proves propositions concerning bags. HOL's bag theory was insufficient for my project, so I have extended it with 25 additional results, which have been merged into HOL for others to use if they wish (see 6 for a list).

Definition 2.4.1. The function bag: set $\mapsto$ bag converts sets into bags. This should not be confused with the type bag: $\alpha \mapsto$ num.

Definition 2.4.2. The function set: bag $\mapsto$ set converts bags into sets. Again, not to be confused with the type set: $\alpha \mapsto$ bool

Definition 2.4.3. The function unibag: bag $\mapsto$ bag converts bags into sets and then back again.

Notation 2.4.4. $b\ e$ is the number of occurrences of the element e in the bag b.

Definition 2.4.5. A bag is distinct if no elements occur more than once. $\vdash$ distinct $b \iff \forall e.\, b\ e \leq 1$

Corollary 2.4.6 (Unibags are distinct)**.** $\vdash \mathsf{distinct}\,(\mathsf{unibag}\,b)$

I needed unibags in order to reason about contraction of hypotheses in **G**. To make a bag of hypotheses a unibag is to make them equivalent to a set of hypotheses, which is necessary in the equivalence proof to come later. The main result concerning unibags was the following:

Theorem 2.4.7 (Complete contraction)**.** $\vdash \Gamma \vdash_{\mathbf{G}} A \iff \mathsf{unibag}\,\Gamma \vdash_{\mathbf{G}} A$

Proof. (if) By **G** weakening.

(only if) By induction on the cardinality of Γ, then an application of the cont (contraction) rule. □

Definition 2.4.8. The merge of two bags, denoted $b \sqcup c$ is the pointwise maximum of the element counts.

$$\vdash b_1 \sqcup b_2 = (\lambda x.\, \mathbf{if}\ b_1\, x < b_2\, x\ \mathbf{then}\ b_2\, x\ \mathbf{else}\ b_1\, x)$$

Lemma 2.4.9 (Bag of set union)**.** *When applied to a set union,* bag *returns a bag merge with the* bag *applied to each set.*

$$\vdash \mathsf{bag}\,(b \cup b') = \mathsf{bag}\,b \sqcup \mathsf{bag}\,b'$$

2.5 Proof of Equivalence

Notation 2.5.1. When I give the HOL tactics of my proofs, I will present them as they are typed in my HOL code. HOL doesn't remember how something is proved, so it can't give a typeset version that I can use. The main thing to note is that the deducibility relations ($\vdash_{\mathbf{G}}$) and ($\vdash_{\mathbf{N}}$) are given in prefix from in my code, and are just typed `G` and `N`. I explain other differences with the proofs.

Lemma 2.5.2 (**G** superset weakening)**.** *Since weakening is built into the axiom of* ***G****, I have proved a lemma to use as a weakening rule. The hypotheses of a sequent can be extended to any finite superset of those hypotheses.*

$$\vdash \Gamma \vdash_{\mathbf{G}} A \Rightarrow \forall \Gamma'.\, \Gamma \leq \Gamma' \wedge \mathsf{finite}\,\Gamma' \Rightarrow \Gamma' \vdash_{\mathbf{G}} A$$

The following two lemmata form the main part of my formalisation, and are used together to prove the main theorem.

Lemma 2.5.3 (From **N** to **G**)**.**

$$\vdash D \vdash_{\mathbf{N}} A \Rightarrow \mathsf{bag}\,D \vdash_{\mathbf{G}} A$$

Proof. The proof is by rule induction on **N**. Given a instance of an inference rule in **N**, I must construct a proof in **G** with the same hypotheses and conclusion.

Three cases are proven by a single rewrite with the rules of **G**, those corresponding to **N** rules: ax, ⩔il and ⩔ir. The introduction rule cases of **N** generally are translated into the right rules of **G**. The elimination rules translate to an instance of the corresponding left rule, plus an instance of cut.

Here is the construction for the ⊥e case, the next shortest case:

$$\dfrac{\dfrac{\vdots}{\mathsf{bag}\,D \vdash_{\mathbf{G}} \bot}\,(\mathrm{IH}) \qquad \dfrac{}{\{\!|\bot|\!\} \vdash_{\mathbf{G}} A}\,\mathsf{L}\bot}{\mathsf{bag}\,D \vdash_{\mathbf{G}} A}\,\mathsf{cut}$$

Here are the HOL tactics which prove this case:

```
‘G {|Bot|} A‘ by metis_tac[G_bot,BAG_IN_BAG_INSERT,FINITE_BAG] >>
metis_tac[G_cut,BAG_UNION_EMPTY]
```

`metis_tac` is the first order reasoner, which takes a `thm list`, where a `thm` is the datatype of propositions which have been proved in HOL. `Bot` is $\bot$, thms starting with `G_` are the names of **G** rules. `BAG_IN_BAG_INSERT` proves that $\bot \in \{\!|\bot|\!\}$, `FINITE_BAG` proves that the hypotheses are finite and `BAG_UNION_EMPTY` removes the empty bag which `cut` introduces.

While in the previous construction matches quite closely with the informal proof, for some cases I used tactics which are less similar in appearance.

Consider the construction of the $\rightarrow$i case:

$$\dfrac{\dfrac{}{\mathsf{bag}\,(\{A\} \cup D) \vdash_{\mathbf{G}} B}\,(\mathrm{IH})}{\mathsf{bag}\,D \vdash_{\mathbf{G}} A \rightarrow B}\,\mathsf{R}{\rightarrow}$$

Here are the HOL tactics for the $\rightarrow$i case:

```
irule G_rimp >>
fs[BAG_OF_SET_INSERT] >>
irule G_lw >>
simp[] >>
drule G_FINITE >>
rw[] >>
qexists_tac ‘BAG_MERGE {|A|} (BAG_OF_SET D)‘ >>
simp[BAG_MERGE_ELBAG_SUB_BAG_INSERT]
```

These tactics are in a backwards-proof style. `irule` reduces the goal (conclusion) to the antecedent of the supplied `thm`. `BAG_OF_SET_INSERT` is an instance of bag of set union limited to singleton union. `G_lw` is **G** weakening. `drule` uses an assumption which matches the antecedent of the supplied `thm` and introduces the conclusion of that thm as an antecedent to the goal, this is like modes ponens. `rw` aggressively rewrites the goal using known rewrite rules, plus any `thms` provided (none here). `qexists_tac` supplies a witness to an existential goal. `simp` rewrites the goal using known rewrites and supplied thms. `BAG_MERGE_ELBAG_SUB_BAG_INSERT` says that a bag merge of a singleton and a bag is a sub-bag of a bag union of a singleton and a bag (so that I can weaken from merge to union).

The following is the case for $\rightarrow$e, first in mathematical notation, then in HOL tactics, which this time display a mostly forwards-proof style. Each line with a `by` derives an assumption from other assumptions and the provided tactics:

$$\dfrac{\dfrac{\dfrac{}{\mathsf{bag}\,D \vdash_{\mathbf{G}} A \rightarrow B}\,\mathrm{IH} \quad \dfrac{\dfrac{}{\mathsf{bag}\,D' \vdash_{\mathbf{G}} A}\,\mathrm{IH} \quad \dfrac{}{\{\!|B|\!\} \vdash_{\mathbf{G}} B}\,\mathsf{ax}}{\{\!|A \rightarrow B|\!\} \uplus D' \vdash_{\mathbf{G}} B}\,\mathrm{L}{\rightarrow}}{\mathsf{bag}\,D \uplus \mathsf{bag}\,D' \vdash_{\mathbf{G}} B}\,\mathsf{cut}}{\mathsf{bag}\,D \sqcup \mathsf{bag}\,D' \vdash_{\mathbf{G}} B}\,(\text{complete contraction})$$

```
rename[‘N D (A Imp B)‘] >>
 simp[BAG_OF_SET_UNION] >>
‘FINITE_BAG (BAG_OF_SET D’)‘ by metis_tac[N_FINITE,FINITE_BAG_OF_SET] >>
‘G (BAG_INSERT B (BAG_OF_SET D’)) B‘
  by simp[G_ax,BAG_IN_BAG_INSERT] >>
‘G (BAG_INSERT (A Imp B) (BAG_OF_SET D’)) B‘
  by metis_tac[G_limp] >>
‘G ((BAG_OF_SET D) + (BAG_OF_SET D’)) B‘
  by metis_tac[G_cut] >>
‘G (unibag (BAG_OF_SET D + BAG_OF_SET D’)) B‘ by metis_tac[G_unibag] >>
fs[unibag_UNION]
```

`rename` changes the name of variables, in this case I renamed `A'` to `B`. The symbol `+` is ASCII for $\uplus$. The first `simp` rewrites the goal as a bag merge, and the final `fs` rewrites the assumption from the previous line as a bag merge, which equals the goal.

I will not go over the remainder of the cases, they are similar in structure to the presented cases. □

Lemma 2.5.4 (From **G** to **N**)**.**

$$\vdash \Gamma \vdash_{\mathbf{G}} A \Rightarrow \mathsf{set}\,\Gamma \vdash_{\mathbf{N}} A$$

Proof. The proof is by rule induction on **G**. Troelstra and Schwichtenberg say that "at each step in the proof we show how to construct from a G-deduction of $\Gamma \Rightarrow A$ an N-deduction of $\Gamma' \Rightarrow A$ for some Γ' with $\Gamma' \subset \mathrm{Set}(\Gamma)$" (Troelstra and Schwichtenberg 2000, p. 69)[2], but I found that you can avoid the need for a subset, by using weakening in the cases which would take a subset in their proof.

Two cases are proven by a single rewrite with the rules of **N**, the two instances of L$\veebar$. The contraction (cont) rule is proven with the tactic `fs[SET_OF_BAG,BAG_UNION,BAG_INSERT]`, which rewrites the goal and assumptions with known rewrites and the supplied `thms`. Here the extra formula disappears when converted to a set.

Since I use weakening rather than some subset of hypotheses, it takes longer to prove the ax and L$\bot$ cases than in the book. In the book they say these rules correspond to proof trees in **N** with a single node A and $\frac{\bot}{A}$ respectively. My tactics to prove the ax rule are as follows:

```
‘?b. Γ = BAG_INSERT A b‘ by metis_tac[BAG_DECOMPOSE] >>
fs[] >>
simp[SET_OF_BAG_INSERT, Once INSERT_SING_UNION] >>
‘N {A} A‘ by metis_tac[N_ax] >>
simp[UNION_COMM] >>
irule N_lw_SUBSET >>
conj_tac >- metis_tac[FINITE_UNION,FINITE_SET_OF_BAG,FINITE_DEF] >>
metis_tac[SUBSET_UNION]
```

The `?` is ASCII for $\exists$. Since I use bag membership in the definition of ax, I must first decompose Γ into an insert. I than replace the occurrences of Γ in the goal and assumptions with the insert expression using `fs`. The `simp` rewrites the goal with the singleton set outside the `SET_OF_BAG` rather than a singleton bag inside. I then use ax to instantiate a proof of A. I then weaken this to a proof of the goal, by rewriting the goal (`simp`) with commutativity, and proving that the goal is a finite superbag of {A} with the last two tactics. `conj_tac` splits a conjunctive goal into two sub-goals, the first of which is the finiteness of the superbag, which I prove with the relevant finiteness lemmata. The second sub-goal is that {A} is a subset of the goal (a union containing {A}), and is solved with the `SUBSET_UNION` lemma.

The right rules of **G** correspond to introduction rules in **N**.

Here is the prooftree for the R$\rightarrow$ case:

$$\dfrac{\dfrac{\dfrac{}{\mathsf{set}\,(\{\!|A|\!\} \uplus \Gamma) \vdash_{\mathbf{N}} B}\,(\mathrm{IH})}{\{A\} \cup \mathsf{set}\,\Gamma \vdash_{\mathbf{N}} B}\,(\text{bring } A \text{ out})}{\mathsf{set}\,\Gamma \vdash_{\mathbf{N}} A \rightarrow B}\,{\rightarrow}\mathsf{i}$$

Here are the tactics which prove the R$\rightarrow$ case:

[2] They actually write $\mathrm{Set}(\Gamma') \subset \Gamma$, but this is incorrect since Γ is the multiset and Γ' is the set.

```
fs[SET_OF_BAG_INSERT] >>
metis_tac[N_impi]
```

The left rules require the assumptions be replaced with an elimination rule which derives the assumption. Here is the case for $\mathsf{L}\veebar$, which differs from the book due to my weakening use as described earlier:

$$\dfrac{\dfrac{\dfrac{}{\mathsf{set}\,(\{\!|A|\!\} \uplus \Gamma) \vdash_{\mathbf{N}} C}\,(\mathrm{IH})}{\mathsf{set}\,\Gamma \setminus \{A\} \vdash_{\mathbf{N}} A \to C}\,{\to}\mathsf{i}\ (\textbf{Nd}\ \text{version}) \qquad \dfrac{\dfrac{}{\{A \barwedge B\} \vdash_{\mathbf{N}} A \barwedge B}\,\mathsf{ax}}{\{A \barwedge B\} \vdash_{\mathbf{N}} A}\,\barwedge\mathsf{e}}{\mathsf{set}\,\Gamma \setminus \{A\} \cup \{A \barwedge B\} \vdash_{\mathbf{N}} C}\,{\to}\mathsf{e}}{\{A \barwedge B\} \cup \mathsf{set}\,\Gamma \vdash_{\mathbf{N}} C}\ (\text{superset weakening, commutativity})$$

Here are the HOL tactics which prove the case, the structure is quite similar to the proof-tree, with some rewrites interspersed, and the last four lines correspond to the last inference of the tree:

```
rename [‘N _ C‘] >>
fs[SET_OF_BAG_INSERT] >>
‘N A And B (A And B)‘ by metis_tac[N_ax] >>
‘N A And B A‘ by metis_tac[N_andel] >>
‘N ((A INSERT (SET_OF_BAG Γ)) DELETE A) (A Imp C)‘
  by metis_tac[N_impi_DELETE] >>
fs[DELETE_DEF] >>
‘N (((SET_OF_BAG Γ) DIFF A) UNION A And B) C‘ by metis_tac[N_impe] >>
‘N ((A And B) INSERT ((SET_OF_BAG Γ) DIFF A)) C‘
          by metis_tac[UNION_COMM,INSERT_SING_UNION] >>
irule N_lw_SUBSET >>
conj_tac >- metis_tac[N_FINITE,FINITE_INSERT] >>
qexists_tac ‘(A And B) INSERT SET_OF_BAG Γ DIFF A‘ >>
rw[SUBSET_DEF]
```

I will not show any of the other cases, as they are all fairly similar.

□

The following is the primary theorem of this project:

Theorem 2.5.5 (Proof of equivalence between **N** and **G**). *Given the same hypotheses, modulo weakening, the same formulae are provable in both calculi. This is Theorem 3.3.1 in Troelstra and Schwichtenberg (2000).*

$$\vdash \Gamma \vdash_{\mathbf{G}} A \iff \mathsf{set}\,\Gamma \vdash_{\mathbf{N}} A$$

Proof. *(only if)* by lemma 2.5.4.

(if) by lemma 2.5.3 and theorem 2.4.7 (Complete contraction).

In HOL:

```
rw[G_N] >>
EQ_TAC >- rw[G_N] >>
rw[] >>
‘G (unibag Γ) A‘ by metis_tac[N_G] >>
```

```
metis_tac[G_unibag]
```

□

3 Discussion of Issues

3.1 Learning Curve

I found HOL to have quite a steep learning curve. Several weeks were dedicated to learning how to prove basic propositions which are trivial to prove on paper. Knowing a proof does not help much if you do not know how to use the theorem prover. I found that the documentation was difficult to read, as it is very technical once you get past the tutorial. However, I now have gained some confidence in HOL, and its particularities make more sense now that I am used to them.

The difficulty lies not just in understanding how HOL works, but also in remembering and knowing how to find the tools that will prove your proposition. HOL implements *goal directed* proof as a method of proving theorems, and this is what I used to prove all of the results in this project. To prove a goal in HOL, one uses *tactics*, which help to construct a proof starting with the desired conclusion. There are many tactics available in HOL, most of which I have not used and don't understand. In addition there is a library of theories which contain theorems which can be used by some tactics as lemmata to advance towards the goal. The combined number of options is intimidating at first, but I found that only a small subset of these tools were necessary for the purposes of this project.

3.2 Bag Theory

A significant portion of effort in this project was dedicated to proving lemmata concerning bags. As discussed earlier, the existing bag theory in HOL contained only some of the results which I needed, so I had to prove them myself. The bag theory in HOL is also somewhat confusing in its formulation. For example, there is a ternary relation called BAG_DELETE which is defined $\mathsf{BAG\_DELETE}\, b_0\, e\, b \iff (b_0 = \mathsf{BAG\_INSERT}\, e\, b)$, in contrast to the set theory binary relation called DELETE which is defined $s\, \mathsf{DELETE}\, x = s \setminus \{x\}$. The first is used to relate two bags, one of which has already had an element deleted, the second (more intuitively) is a function which deletes an element from a set.

In the process of proving the necessary bag lemmata, I often found myself looking at a mess of conditionals inside lambda abstractions. For a simple example, suppose I wanted to prove that $\mathsf{bag}\, s \setminus b = \mathsf{bag}\, (s \setminus \mathsf{set}\, b)$ and I expand the definitions of the operators, I get:

$$(\lambda x.\, (\mathbf{if}\, s\, x\, \mathbf{then}\, 1\, \mathbf{else}\, 0) \setminus b\, x) =$$
$$(\lambda x.\, \mathbf{if}\, s\, x \wedge \neg(b\, x \geq 1)\, \mathbf{then}\, 1\, \mathbf{else}\, 0).$$

The only way I found to prove a goal like this is to use $\mathsf{FUN\_EQ\_THM}: \vdash (f = g) \iff \forall x.\, f\, x = g\, x$, a theorem which does not mention lambda abstractions nor conditionals, so took some time for me to find.

3.3 Summary of effort

The effort and time required to formalise mathematics in HOL is more than that which it takes to cover the same content informally. After a few weeks learning to use HOL, another full month has been spent formalising content which I understood after only a day or two of reading Troelstra and Schwichtenberg (2000). In total I have spent over 40 hours learning HOL, and 90 hours formalising the relevant proof theory.

4 Future Work

4.1 Extensions of the proof

4.1.1 Classical Logic

I would have liked to have formalised the proof for classical logic as well as intuitionistic logic, but I was unable to due to time. I did make some progress, but due to the differences between the classical absurdity rules of Natural Deduction and Sequent Calculus, extra work is required which I have not completed. The rest of the rules did not seem to cause any significant difficulty, and in fact I was able to prove all cases of $\mathbf{N} \Rightarrow \mathbf{G}$, except for the negation case, in a single afternoon.

Here are the negation rules in question. Note that in classical sequent calculus the consequent is a bag of formulae rather than an individual formula:

$$\frac{\{\neg A\} \cup D \vdash_{\mathbf{N}} \bot}{D \vdash_{\mathbf{N}} A}\ \bot_c\mathsf{e} \qquad \frac{\bot \in \Gamma \quad \mathsf{finite}\,\Gamma \quad \mathsf{finite}\,\Delta}{\Gamma \vdash_{\mathbf{G}} \Delta}\ \mathsf{L}\bot_c$$

Troelstra and Schwichtenberg leave the proof of equivalence for classical logic as an exercise for the reader (Troelstra and Schwichtenberg 2000, Thm. 3.3.3).

4.1.2 First Order Logic

I would have liked to have extended the proof to first order logic. In future it would be interesting to do so. The first order proof contains the propositional one I have done, and they do not separate them in Troelstra and Schwichtenberg (ibid., Sec. 3.3).

4.1.3 Cut-free proofs and Normalisation

A substantial part of Troelstra and Schwichtenberg (ibid.) is dedicated to cut-free sequent calculus (**G** without the cut rule), and there is a proof that this is equivalent to normalised natural deduction (ibid., Sec. 6.3.1). Cut-free sequent calculus is interesting because it has the subformula property, that is, in any proof of $\Gamma \vdash_{\mathbf{G}} \Delta$, only subformulae of Γ and Δ appear. This has many applications, for example, propositional intuitionistic logic is decidable, and a decision procedure exists in a cut-free sequent calculus (ibid., Thm. 4.2.6).

4.2 Other Proof Theory

Other calculi of interest to me are those for modal logics. Troelstra and Schwichtenberg present a sequent calculus for **S4**, and prove that intuitionistic logic can be embedded into it (ibid., Sec. 9.2). This would have been more interesting to formalise, but I expect would have been more challenging as I have less experience with calculi of non-classical logics. There are many modal logics and multiple calculi for each, so there is a room for more original formalisation in this area.

4.3 Flexible sets of rules

The main improvement I would have liked to have made to my formalisation is a significant alteration in how the proof systems are defined. Since there are many variants of the proof systems, and many of these build upon each other, it would be optimal to be able to specify a set of inference rules rather than redefine all of the redundant rules each time I want to define a deducibility relation for a system.

This would allow me to define a set of cut-free sequent calculus rules, and then add the cut rule to them to form cut-full sequent calculus, for example.

This modification to my formalisation would make the other extensions I considered easier to implement, and would also decrease clutter in the HOL code.

5 Conclusion

I have successfully mechanised the equivalence proof between propositional intuitionistic natural deduction and sequent calculus. In doing so I have learnt a lot about theorem proving and have deepened my understanding of proof theory and mathematical logic, thus satisfying the purpose of undertaking this project. While learning to use HOL was a challenge, the tools which HOL has for inductive definitions and rule induction seem well suited to formalisation of proof theory. I now have an appreciation for the significant effort required to formalise mathematics in a theorem prover compared to proving the same mathematics on paper, and have come to enjoy this process nonetheless.

5.1 Acknowledgements

I am very grateful to Michael Norrish for supervising my project, AMSI for awarding me a Vacation Research Scholarship for this project and my wife Myvanwy for listening to my logic-fuelled rants.

References

Camilleri, Juanito and Tom Melham (Aug. 1992). *Reasoning with inductively defined relations in the HOL theorem prover*. Tech. rep. UCAM-CL-TR-265. University of Cambridge, Computer Laboratory.

Doorn, Floris van (2015). "Propositional Calculus in Coq". In: *arXiv preprint arXiv:1503.08744*.

Mikhajlova, Anna and Joakim von Wright (1998). "Proving isomorphism of first-order logic proof systems in HOL". In: *International Conference on Theorem Proving in Higher Order Logics*. Springer, pp. 295–314.

Slind, Konrad and Michael Norrish (2008). "A brief overview of HOL4". In: *International Conference on Theorem Proving in Higher Order Logics*. Springer, pp. 28–32.

Troelstra, Anne Sjerp and Helmut Schwichtenberg (2000). *Basic proof theory*. Cambridge University Press.

6 Appendix

6.1 Bag Lemmata

This is a list of the theorems I have formalised in HOL. The source for this project can be found at `https://github.com/lxndrcx/proofTheoryHOL`. In addition, the following bag and unibag lemmata I wrote have been merged into HOL, see `https://github.com/HOL-Theorem-Prover/HOL/pull/654`. I have given the theorem names as they appear in HOL.

Lemma 6.1.1 (BAG_MERGE_SUB_BAG_UNION). $\vdash s \sqcup t \leq s \uplus t$

Lemma 6.1.2 (BAG_MERGE_EMPTY). $\vdash (\{\!|\}\!| \sqcup b = b) \wedge (b \sqcup \{\!|\}\!| = b)$

Lemma 6.1.3 (BAG_MERGE_ELBAG_SUB_BAG_INSERT). $\vdash \{\!|A|\!\} \sqcup b \leq \{\!|A|\!\} \uplus b$

Lemma 6.1.4 (BAG_MERGE_EQ_EMPTY). $\vdash (a \sqcup b = \{\!|\}\!|) \iff (a = \{\!|\}\!|) \wedge (b = \{\!|\}\!|)$

Lemma 6.1.5 (BAG_INSERT_EQ_MERGE_DIFF).

$$\vdash (\{\!|e|\!\} \uplus a = b \sqcup c) \Rightarrow (b \sqcup c = \{\!|e|\!\} \uplus (b \setminus \{\!|e|\!\} \sqcup (c \setminus \{\!|e|\!\})))$$

Lemma 6.1.6 (BAG_MERGE_BAG_INSERT).

$$\begin{array}{l}\vdash (a\,e \leq b\,e \Rightarrow (a \sqcup (\{\!|e|\!\} \uplus b) = \{\!|e|\!\} \uplus (a \sqcup b))) \wedge \\ (b\,e < a\,e \Rightarrow (a \sqcup (\{\!|e|\!\} \uplus b) = a \sqcup b)) \wedge \\ (a\,e < b\,e \Rightarrow (\{\!|e|\!\} \uplus a \sqcup b = a \sqcup b)) \wedge \\ (b\,e \leq a\,e \Rightarrow (\{\!|e|\!\} \uplus a \sqcup b = \{\!|e|\!\} \uplus (a \sqcup b))) \wedge \\ ((a\,e = b\,e) \Rightarrow (\{\!|e|\!\} \uplus a \sqcup (\{\!|e|\!\} \uplus b) = \{\!|e|\!\} \uplus (a \sqcup b)))\end{array}$$

Lemma 6.1.7 (BAG_OF_SET_UNION). $\vdash \mathsf{bag}\,(b \cup b') = \mathsf{bag}\,b \sqcup \mathsf{bag}\,b'$

Lemma 6.1.8 (BAG_OF_SET_INSERT). $\vdash \mathsf{bag}\,(\{e\} \cup s) = \{\!|e|\!\} \sqcup \mathsf{bag}\,s$

Lemma 6.1.9 (BAG_OF_SET_BAG_DIFF_DIFF). $\vdash \mathsf{bag}\,s \setminus b = \mathsf{bag}\,(s \setminus \mathsf{set}\,b)$

Lemma 6.1.10 (SET_OF_EL_BAG). $\vdash \mathsf{set}\,\{\!|e|\!\} = \{e\}$

Lemma 6.1.11 (BAG_OF_SET_EQ_INSERT). $\vdash (\{\!|e|\!\} \uplus b = \mathsf{bag}\,s) \Rightarrow \exists s'.\, s = \{e\} \cup s'$

Lemma 6.1.12 (FINITE_BAG_MERGE). $\vdash \mathsf{finite}\,(a \sqcup b) \iff \mathsf{finite}\,a \wedge \mathsf{finite}\,b$

Lemma 6.1.13 (BAG_MERGE_CARD).

$$\begin{array}{l}\vdash \mathsf{finite}\,a \wedge \mathsf{finite}\,b \Rightarrow \\ \mathsf{cardinality}\,(a \sqcup b) \leq \mathsf{cardinality}\,a + \mathsf{cardinality}\,b\end{array}$$

Lemma 6.1.14 (BAG_ALL_DISTINCT_SUB_BAG). $\vdash s \leq t \wedge \mathsf{distinct}\,t \Rightarrow \mathsf{distinct}\,s$

Definition 6.1.15. $\mathsf{filter}\,P\,b$ returns b filtered to include only elements of P:[3]

$$\vdash \mathsf{filter}\,P\,b = (\lambda\,e.\,\mathbf{if}\,P\,e\,\mathbf{then}\,b\,e\,\mathbf{else}\,0)$$

Lemma 6.1.16 (BAG_OF_SET_DIFF). $\vdash \mathsf{bag}\,(s \setminus s') = \mathsf{filter}\,(\mathsf{complement}\,s')\,(\mathsf{bag}\,s)$

Lemma 6.1.17 (FINITE_BAG_OF_SET). $\vdash \mathsf{finite}\,(\mathsf{bag}\,s) \iff \mathsf{finite}\,s$

[3]Not my definition, just included to explain next lemma

6.2 Unibag Lemmata

Lemma 6.2.1 (unibag_INSERT). $\vdash \mathsf{unibag}\,(\{\!|a|\!\} \uplus b) = \{\!|a|\!\} \sqcup \mathsf{unibag}\,b$

Lemma 6.2.2 (unibag_UNION). $\vdash \mathsf{unibag}\,(a \uplus b) = \mathsf{unibag}\,a \sqcup \mathsf{unibag}\,b$

Lemma 6.2.3 (BAG_IN_unibag). $\vdash e \in \mathsf{unibag}\,b \iff e \in b$

Lemma 6.2.4 (unibag_EQ_BAG_INSERT). $\vdash (\mathsf{unibag}\,b = \{\!|e|\!\} \uplus b') \Rightarrow \exists\, c.\, b' = \mathsf{unibag}\,c$

Lemma 6.2.5 (unibag_FINITE). $\vdash \mathsf{finite}\,(\mathsf{unibag}\,b) \iff \mathsf{finite}\,b$

Lemma 6.2.6 (unibag_ALL_DISTINCT). $\vdash \mathsf{distinct}\,(\mathsf{unibag}\,b)$

Lemma 6.2.7 (unibag_EL_MERGE_cases).

$$\vdash (e \in b \Rightarrow (\{\!|e|\!\} \sqcup \mathsf{unibag}\,b = \mathsf{unibag}\,b)) \wedge \\ (\neg(e \in b) \Rightarrow (\{\!|e|\!\} \sqcup \mathsf{unibag}\,b = \{\!|e|\!\} \uplus \mathsf{unibag}\,b))$$

Lemma 6.2.8 (unibag_DECOMPOSE). $\vdash \mathsf{unibag}\,g \neq g \Rightarrow \exists\, A\, g_0.\, g = \{\!|A;\, A|\!\} \uplus g_0$

Lemma 6.2.9 (unibag_SUB_BAG). $\vdash \mathsf{unibag}\,b \leq b$

6.3 Main Lemmata and Theorems

Lemma 6.3.1 (N_FINITE). $\vdash D \vdash_{\mathsf{N}} A \Rightarrow \mathsf{finite}\,D$

Lemma 6.3.2 (N_lw). $\vdash D \vdash_{\mathsf{N}} A \Rightarrow \forall B.\, \{B\} \cup D \vdash_{\mathsf{N}} A$

Lemma 6.3.3 (Nd_lw). $\vdash D \vdash_{\mathsf{Nd}} A \Rightarrow \forall B.\, \{B\} \cup D \vdash_{\mathsf{Nd}} A$

Lemma 6.3.4 (N_lw_SUBSET). $\vdash \mathsf{finite}\,D' \Rightarrow \forall D\, A.\, D \vdash_{\mathsf{N}} A \wedge D \subseteq D' \Rightarrow D' \vdash_{\mathsf{N}} A$

Lemma 6.3.5 (Nd_lw_SUBSET). $\vdash \mathsf{finite}\,D' \Rightarrow \forall D\, A.\, D \vdash_{\mathsf{Nd}} A \wedge D \subseteq D' \Rightarrow D' \vdash_{\mathsf{Nd}} A$

Lemma 6.3.6 (N_impi_DELETE). $\vdash D \vdash_{\mathsf{N}} A \Rightarrow D \setminus \{B\} \vdash_{\mathsf{N}} B \to A$

Theorem 6.3.7 (N_Nd). $\vdash D \vdash_{\mathsf{N}} A \iff D \vdash_{\mathsf{Nd}} A$

Lemma 6.3.8 (G_FINITE). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \mathsf{finite}\,\Gamma$

Lemma 6.3.9 (G_lw). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \forall \Gamma'.\, \Gamma \leq \Gamma' \wedge \mathsf{finite}\,\Gamma' \Rightarrow \Gamma' \vdash_{\mathsf{G}} A$

Lemma 6.3.10 (G_lw_BAG_INSERT). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \forall B.\, \{\!|B|\!\} \uplus \Gamma \vdash_{\mathsf{G}} A$

Lemma 6.3.11 (G_lw_BAG_MERGE). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \forall \Gamma'.\, \mathsf{finite}\,\Gamma' \Rightarrow \Gamma' \sqcup \Gamma \vdash_{\mathsf{G}} A$

Lemma 6.3.12 (G_lw_BAG_UNION). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \forall \Gamma'.\, \mathsf{finite}\,\Gamma' \Rightarrow \Gamma \uplus \Gamma' \vdash_{\mathsf{G}} A$

Lemma 6.3.13 (G_unibag). $\vdash \Gamma \vdash_{\mathsf{G}} A \iff \mathsf{unibag}\,\Gamma \vdash_{\mathsf{G}} A$

Lemma 6.3.14 (N_G). $\vdash D \vdash_{\mathsf{N}} A \Rightarrow \mathsf{bag}\,D \vdash_{\mathsf{G}} A$

Lemma 6.3.15 (G_N). $\vdash \Gamma \vdash_{\mathsf{G}} A \Rightarrow \mathsf{set}\,\Gamma \vdash_{\mathsf{N}} A$

Theorem 6.3.16 (G_iff_N). $\vdash \Gamma \vdash_{\mathsf{G}} A \iff \mathsf{set}\,\Gamma \vdash_{\mathsf{N}} A$