Abstract

Abstract

Contents

1 Introduction

2 Agda proof assistant

3 Propositional calculus in Agda

Propositional calculus is a formal system that consists of a set of propositional constants, symbols, inference rules, and axioms. The symbols in propositional calculus represent logical connectives and parenthesis.

The inference rules of propositional calculus define how these symbols can be used. These inference rules specify how to construct well-formed formulas that follow the syntax of the system.

The axioms of propositional calculus are the initial statements or assumptions from which we can derive additional statements using the inference rules.

The semantics of propositional calculus define how the expressions in the system correspond to truth values, typically "true" or "false".

3.1 Formulas

Definition 3.1 (Language)**.** *The language $\mathcal{L}$ of propositional calculus consists of*

- *proposition symbols:* $p_0, p_1, \ldots, p_n$*,*
- *logical connectives:* $\wedge, \vee, \neg, \top, \bot$*,*
- *auxiliary symbols: (,).*

Note that we have omitted the common logical connectives $\to$ and $\leftrightarrow$. This is becuase we can define them using other connectives,

$$\phi \to \psi \stackrel{\mathsf{def}}{=} \neg\phi \vee \psi,$$
$$\phi \leftrightarrow \psi \stackrel{\mathsf{def}}{=} (\neg\phi \vee \psi) \wedge (\neg\psi \vee \phi),$$

making them reduntant. It is possible to choose an even smaller set of connectives [1], but we choose this as it is convenient.

Definition 3.2 (Well formed formula)**.** *The set of well formed formulas is inductively defined as*

- *any propositional constant* $p_0, p_1, \ldots, p_n$ *is a well formed formula,*
- $\top$ *and* $\bot$ *are well formed formulas,*
- *if* p *is a well formed formula, then so is*

$$\neg p,$$

- *if p_i and p_j are well formed formulas, then so are*

$$p_i \wedge p_j \quad \textit{and} \quad p_i \vee p_j.$$

The formula $\top$ should be thought of as the proposition that is always true, and the formula $\bot$ interpreted as the proposition that is always false.

We can represent the concept of a well formed formula in `Agda` as a Type

```
data Formula : Type where
  _∧_ : Formula → Formula → Formula
  _∨_ : Formula → Formula → Formula
  ¬_  : Formula → Formula
  const : ℕ → Formula
  ⊥   : Formula
  ⊤   : Formula
```

3.2 Context

Definition 3.3 (Context)**.** *A set of sentences in the language $\mathcal{L}$. The set is defined inductively as*

- *the empty set is a context*
- *if Γ is a context, then $\Gamma \cup \{\phi\}$ is also a context, where ϕ a formula.*

In `Agda` we can define a context Type

```
data ctxt : Type where
  ∅  : ctxt
  _:_ : ctxt → Formula → ctxt
```

We also need a way to determine if a given formula is in a given context.

Definition 3.4 (Lookup)**.** *For all contexts Γ and all formulas ϕ and ψ*

- $\phi \in \Gamma \cup \{\phi\}$,
- *if $\phi \in \Gamma$, then $\phi \in \Gamma \cup \{\psi\}$.*

We represent this as a Typein `Agda`

```
data _∈_ : Formula → ctxt → Type where
  Z : ∀ {Γ φ} → φ ∈ (Γ : φ)
  S_ : ∀ {Γ φ ψ} → φ ∈ Γ → φ ∈ (Γ : ψ)
```

3.3 Inference rules

3.3.1 Conjunction

3.3.2 Disjunction

3.3.3 Negation

3.3.4 Rules for $\top$ and $\bot$

3.3.5 Axiom

3.3.6 Law of excluded middle

3.3.7 Structural rules

```
data _⊢_ : ctxt → Formula → Type where
  ∧-intro : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ φ → Γ ⊢ ψ → Γ ⊢ φ ∧ ψ
  ∧-elimˡ : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ φ ∧ ψ → Γ ⊢ φ
  ∧-elimʳ : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ φ ∧ ψ → Γ ⊢ ψ
  ∨-introˡ : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ ψ → Γ ⊢ φ ∨ ψ
  ∨-introʳ : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ φ → Γ ⊢ φ ∨ ψ
  ∨-elim : {Γ : ctxt} (φ ψ γ : Formula) → Γ ⊢ φ ∨ ψ → (Γ : φ) ⊢ γ → (Γ : ψ) ⊢ γ → Γ ⊢ γ
  ¬-intro : {Γ : ctxt} {φ : Formula} → (Γ : φ) ⊢ ⊥ → Γ ⊢ ¬ φ
  RAA : {Γ : ctxt} {φ : Formula} → (Γ : ¬ φ) ⊢ ⊥ → Γ ⊢ φ
  ⊥-intro : {Γ : ctxt} {φ : Formula} → Γ ⊢ φ ∧ ¬ φ → Γ ⊢ ⊥
  ⊥-elim : {Γ : ctxt} {φ : Formula} → (Γ : ⊥) ⊢ φ
  ⊤-intro : ∅ ⊢ ⊤
  axiom : {Γ : ctxt} {φ : Formula} → φ ∈ Γ → Γ ⊢ φ
  LEM : {Γ : ctxt} {φ : Formula} → Γ ⊢ φ ∨ ¬ φ
  weakening : {Γ : ctxt} {φ ψ : Formula} → Γ ⊢ ψ → (Γ : φ) ⊢ ψ
  exchange : {Γ : ctxt} {φ ψ γ : Formula} → ((Γ : φ) : ψ) ⊢ γ → ((Γ : ψ) : φ) ⊢ γ
  contraction : {Γ : ctxt} {φ ψ : Formula} → ((Γ : φ) : φ) ⊢ ψ → (Γ : φ) ⊢ ψ
```

3.4 Properties of a propositional calculus

4 Lindenbaum-Tarski algebra

[What is LT?]

4.1 Representing Lindenbaum Tarski algebra in Agda

4.2 Proof that the Lindenbaum Tarski algebra is Boolean

4.3 Soundness

References

[1] Dick van Dalen. *Logic and structure.* Springer, fifth edition, 2013.