TT2026 Introduction to Proof Systems notes

Remaining TODOs: 11

Contents

1. Logic

Definition 1.1: Logic is the study of the principles of correct reasoning.

This requires:
- an unambiguous language in which we can formulate statements
- a mathematical framework to determine the truth of a statement

Definition 1.2: A logical syllogism is an example of correct reasoning.

For example:

All beings are mortal;
All humans are beings;
Therefore all humans are mortal.

2. Propositional logic

2.1. Syntax

Definition 2.1.1: Basic sentences in propositional logic are atomic propositions.

For example, a : Alice is an architect.

Definition 2.1.2: Compound sentences can be formed by logical connectives: $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction).

e.g. $\neg a$; $a \vee b$.

Example:

$$\{\neg c, a \vee b, b \to c\} \vDash a$$

"$\vDash$" means "entails"; we have some propositions and we draw a conclusion.

Definition 2.1.3: Let $X = \{x_1, x_2, ...\}$ be a countably infinite set of propositional variables.

Formulas of propositional logic are defined inductively:

1. *(basis clause)* `true`,`false` and every propositional variable x are propositional formulae
3. *(inductive clause)* If F, G are formulae, then so are $\neg F$ (negation), $F \vee G$ (disjunction, with F and G the disjuncts), and $F \wedge G$ (conjunction, with F and G the conjuncts)
5. *(extremal clause)* Nothing else is a formula

Remark: The extremal clause of the above definition ensures that the set of propositional formulae is the minimal set that satisfies the basis and inductive clauses.

Definition 2.1.4: There are some derived connectives:

- implication: $F \to G := \neg F \wedge G$. Here F is the antecedent and G is the consequent
- bi-implication: $F \leftrightarrow G := (F \to G) \wedge (G \to F)$
- Exclusive-OR: $F \oplus G := (F \wedge \neg G) \vee (\neg F \wedge G)$
- Indexed conjunction: $\bigwedge_{i=1}^{n} F_i := (...((F_1 \wedge F_2) \wedge F_3) \wedge ... \wedge F_n)$
- Indexed disjunction: $\bigvee_{i=1}^{n} F_i := (...((F_1 \vee F_2) \vee F_3) \vee ... \vee F_n)$

Definition 2.1.5: The set of all formulae of proposition logic over X is denoted by $\mathcal{F}(X)$.

Definition 2.1.6: The operate precedence for propositional logic is:

$$\bigvee_i, \bigwedge_i \ll \leftrightarrow \ll \to \ll \wedge, \vee \ll \neg$$

Definition 2.1.7: A literal is an atomic proposition or its negation.

Definition 2.1.8: We say that F is in conjunctive normal form (CNF) if

$$F = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j},$$

where each $L_{i,j}$ is a literal.

F is in disjunctive normal form (DNF) if F

$$F = \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m_i} L_{i,j}.$$

Definition 2.1.9: F is in k-CNF if it is in CNF, and all the m_is are equal to some fixed k.

F is in k-DNF if it is in DNF with all the m_is equal to fixed k.

Remark: Normal forms are particularly well-suited for algorithmic treating.

Definition 2.1.10: Functions on formulae of propositional logic are **uniquely** defined by specifying the function values for "base cases" and "inductive steps", following the cases in Definition 2.1.3. Such a definition is said to use structural induction.

Example: Suppose we want to define a function that returns the set of all subformulae of a formula F.

We define $\text{sub} : F(X) \to \mathcal{P}(F(X))$ using structural induction,

$$\begin{aligned}
\text{sub}(\texttt{true}) &:= \{\texttt{true}\} \\
\text{sub}(\texttt{false}) &:= \{\texttt{false}\} \\
\text{sub}(x) &:= \{x\} \\
\text{sub}(\neg F) &:= \text{sub}(F) \cup \{\neg F\} \\
\text{sub}(F \wedge G) &:= \text{sub}(F) \cup \text{sub}(G) \cup \{F \wedge G\} \\
\text{sub}(F \vee G) &:= \text{sub}(F) \cup \text{sub}(G) \cup \{F \vee G\}
\end{aligned}$$

2.2. (Tarski-style) semantics

Definition 2.2.1: An assignment is a function $\mathcal{A} : X \to \{0, 1\}$ that induces a function $\hat{\mathcal{A}} : \mathcal{F}(X) \to \{0, 1\}$. That is $\mathcal{A}$ maps propositional variables to truth values, and $\hat{\mathcal{A}}$ maps formulae to truth values.

We define $\hat{\mathcal{A}}$ using structural induction.

$$\hat{\mathcal{A}}(\texttt{false}) := 0;$$

$$\hat{\mathcal{A}}(\texttt{true}) := 1;$$

$$\hat{\mathcal{A}}(x) := \mathcal{A}(x) \text{ for every } x \in X;$$

$$\hat{\mathcal{A}}(\neg F) := \begin{cases} 1 & \text{if } \hat{\mathcal{A}}(F) = 0 \\ 0 & \text{otherwise;} \end{cases}$$

$$\hat{\mathcal{A}}(F \wedge G) := \begin{cases} 1 & \text{if } \hat{\mathcal{A}}(F) = 1 \text{ and } \hat{\mathcal{A}}(G) = 1 \\ 0 & \text{otherwise;} \end{cases}$$

$$\hat{\mathcal{A}}(F \vee G) := \begin{cases} 1 & \text{if } \hat{\mathcal{A}}(F) = 1 \text{ or } \hat{\mathcal{A}}(G) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Note that we define $\hat{\mathcal{A}}$ in a kind of meta language - the "and" and "or" here are not the same as in propositional logic.

From now we write $\mathcal{A}$ instead of $\hat{\mathcal{A}}$ for convenience.

Remark: We can construct truth tables for logical and derived connectives in the obvious manner. Implication may be surprising:

a	b	$a \to b$
true	true	true
true	false	false
false	false	false
false	true	true

In particular this means that you can use `false` to show anything to be true, i.e. we should start with true assumptions when trying to prove something (obviously).

Definition 2.2.2: Let $F, G \in \mathcal{F}(X), \mathcal{S} \subseteq \mathcal{F}(X)$ and $\mathcal{A} : X \to \{0,1\}$. Then

1. If $\mathcal{A}(F) = 1$ then $\mathcal{A} \vDash F$ ("$\mathcal{A}$ is a model of F", or "F holds under $\mathcal{A}$").
2. If $\mathcal{A} \vDash F$ for all $F \in \mathcal{S}$, then we write $A \vDash S$ ("$\mathcal{A}$ models $\mathcal{S}$")
3. F is satisfiable is there is some $\mathcal{A} : X \to \{0,1\}$ s.t. $\mathcal{A}(F) = 1$, or $\mathcal{A} \vDash F$. Otherwise, F is unsatisfiable.
4. If F holds under any assignment, we say that F is valid, or that F is a tautology.
5. If, for all assignments $\mathcal{A}$, $\mathcal{A} \vDash \mathcal{S}$ implies $\mathcal{A} \vDash F$ for some F (not neccessarily an $F \in \mathcal{S}$ from statement 2), then $\mathcal{S}$ entails F, written $\mathcal{S} \vDash F$. We write $G \vDash F$ if $\{G\} \vDash F$.
6. If $F \vDash G$ and $G \vDash F$, then F and G are logically equivalent, $F \equiv G$.
7. If F is satisfiable iff G is satisfiable, then F and G are equisatisfiable.

Example:

$$\{x, x \to y\} \vDash y$$
$$x \to (y \to z) \equiv (x \wedge y) \to z$$
$$x \vee y \text{ is equisat with } (x \vee z) \wedge (y \vee \neg z)$$

Example (Encoding constraint satisfaction problem - Hamiltonian path problem): For undirected graph $G := (V, E)$, is there a path visiting every vertex exactly once?

Introduce propositional variables $x_{i,j}$ expressing whether a Hamiltonian path visits vertex i at time j in a Hamiltonian path.

$$F_1 := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} x_{i,j}$$
$$F_2 := \bigwedge_{i=1}^{n} \bigwedge_{1 \le j \le k \le n} \neg(x_{i,j} \wedge x_{i,k})$$
$$F_3 := \bigwedge_{i=1}^{n} \bigwedge_{k=1}^{n} \bigwedge_{j=1}^{n-1} x_{i,j} \wedge x_{k,j+1} \to e_{i,k}$$
$$F_4 : \bigwedge_{\{i,j\} \in E} e_{i,j} \wedge \bigwedge_{\{i,j\} \notin E} \neg e_{i,j}.$$

Then $F_1 \wedge F_2 \wedge F_3 \wedge F_4$ iff G has a Hamiltonian path.

This is a practical example of why satisfiable is an important concept.

Remark: This is nonconstructive, and also uses the law of the excluded middle; some claim that using the law of the excluded middle for nonconstructive proofs is invalid.

2.3. Minimal calculus (minimal logic)

Definition 2.3.1: The minimal calculus, $\mathbf{M}_0$ consists of a finite number of axioms:

- $PL_1 : A \to (A \wedge A)$
- $PL_2 : (A \wedge B) \to (B \wedge A)$
- $PL_3 : (A \to B) \to [(A \wedge C) \to (B \wedge C)]$
- $PL_4 : [(A \to B) \to (B \to C)] \to (A \to C)$
- $PL_5 : B \to (A \to B)$
- $PL_6 : (A \wedge (A \to B)) \to B$
- $PL_7 : A \to (A \vee B)$
- $PL_8 : (A \vee B) \to (B \vee A)$
- $PL_9 : [(A \to C) \wedge (B \to C)] \to [(A \vee B) \to C]$
- $PL_{10} : [(A \to B) \wedge (A \to \neg B)] \to \neg A$

We also have a single inference rule, modus ponens: From A and $A \to B$, infer B.

Remark: Modus ponens is basically PL_6, but it allows us to actually perform pattern-matching on the proof and reduce it.

Remark: All of the axioms are also valid formulae in Tarski-style semantics

Remark: Implication needs to be a primitive in the minimal logic, so that we can deal very carefully with negation.

Remark: The subscript 0 in $\mathbf{M}_0$ indicates that this is a propositional logic (as opposed to, e.g., predicate logic)

Definition 2.3.2: A derivation in the minimal calculus is a finite sequence of formulae $A_1, ..., A_n$, each A_i being either an axiom, or obtained from $A_j, A_k, j, k < i$ by application of modus ponens. If some A_i is neither an axiom nor MP-derived, it is a hypothesis.

We write $\vdash_{\mathbf{M}_0} B$ if B is derivable; then B is a theorem or provable in $\mathbf{M}_0$.

Example:

1. $\vdash C$ (hypothesis)
2. $\vdash C \to (D \to C)$ (PL_5)
3. $\vdash D \to C$ $(MP1, 2)$
4. $\vdash (D \to C) \to [(D \to D) \to (C \wedge D)]$ (PL_3)
5. $\vdash (D \wedge D) \to (C \wedge D)$ $(MP3, 4)$
6. $\vdash D \to (D \wedge D)$ (PL_1)
7. $\vdash D$ (hypothesis)
8. $\vdash D \wedge D$ $(MP6, 7)$
9. $\vdash C \wedge D$ $(MP5, 2)$

This gives us a new proof rule, conjunction introduction:

$$\wedge_{\text{INTRO}} : \text{If } \vdash_{\mathbf{M}_0} C \text{ and } \vdash_{\mathbf{M}_0} D \text{ then } \vdash_{\mathbf{M}_0} C \wedge D.$$

Example:

1. $A \to B$ (hypothesis)
2. $B \to C$ (hyp)
3. $(A \to B) \wedge (B \to C)$ ($\wedge_{\text{INTRO}}$ (1, 2))
4. $[(A \to B) \wedge (B \to C)] \to (A \to C)$ (PL4)
5. $A \to C$ (MP (3, 4))

This gives us a new rule, transitivity of implication,

$$\to_{\text{TRANS}}: \text{If } \vdash_{\mathbf{M}_0} A \to B \text{ and } \vdash_{\mathbf{M}_0} B \to C \text{ then } \vdash_{\mathbf{M}_0} A \to C.$$

Definition 2.3.3: Ex falso quodlibet ("from falsehood, anything follows") is the principal that $\neg A \to (A \to B)$, i.e. anything is provable from a false hypothesis.

Definition 2.3.4: Tertium non datur ("no third [possibility] is given") is the law of the excluded middle, $\neg\neg A \to A$ (equivalently, $A \wedge \neg A$ is a tautology).

Remark: Not every valid statement in Tarski-style semantics is derivable using the minimal calculus.

In particular, it is not possible to derive ex falso quodlibet or tertium non datur.

Proposition 2.3.5: $\nvdash_{\mathbf{M}_0} \neg A \to (A \to B)$

Proof: Let $h : \mathcal{F}(X) \to \{0, 1\}$ be such that:

- Propositional and template variables are given an arbitrary but fixed value
- $h(\neg F) = 0$
- The remaining logical connectives are reduced inductively according to the following table

$h(F)$	$h(G)$	$h(F \wedge G)$	$h(F \vee G)$	$h(F \to G)$
0	0	1	0	0
0	1	0	1	1
1	0	0	1	0
1	1	0	1	0

Then choose h such that $h(A) = 0, h(B) = 1$, and observe that all aaxioms of $\mathbf{M}_0$ evaluate to 0 under h, while $h(\neg A \to (A \to B)) = 1$.

Observe also that if $h(A) = h(A \to B) = 0$, then applying modus ponens to get B must have $h(B) = 0$, otherwise $h(A \to B)$ would evaluate to 1 by the definition of h.

Therefore, starting from axioms and using modus ponens, we cannot create a formula F that evaluates to 1 under h; therefore $\neg A \to (A \to B)$ is not derivable from the axioms and modus ponens. □

Definition 2.3.6: If we add ex falso quodlibet as an extra axiom, PL11 $\neg A \to (A \to B)$, then we get intuitionistic logic, denoted $\mathbf{J}_0$.

If we add tertium non datur, the law of the excluded middle, as a 12th axiom, we get classical logic, denoted $\mathbf{K}_0$.

Theorem 2.3.7: For any formula F, $\vDash F$ (F is valid) iff $\vdash_{\mathbf{K}_0} F$.
$\Longleftarrow$ is soundness and is simple.
$\Longrightarrow$ is completeness and is more difficult to prove.

2.4. Equational reasoning

Idea: substitute subformulae by equivalent ones, according to the axioms of Boolean algebras.

Definition 2.4.1: A Boolean algebra is a structure that satisfies the following axioms:

- $A \vee A \equiv A$
- $A \wedge A \equiv A$ (idempotence)
- $A \wedge B \equiv B \wedge A$
- $A \vee B = B \vee A$ (commutativity)
- $(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$

- $(A \vee B) \vee C \equiv A \vee (B \vee C)$ (associativity)
- $A \wedge (A \vee B) \equiv A$
- $A \vee (A \wedge B) \equiv A$ (absorption)
- $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
- $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$ (distributivity)
- $\neg\neg A \equiv A$ (double negation)
- $\neg(A \wedge B) \equiv \neg A \vee \neg B$
- $\neg(A \vee B) \equiv \neg A \wedge \neg B$ (de Morgan)
- $A \vee \neg A \equiv \texttt{true}$
- $A \wedge \neg A \equiv \texttt{false}$ (complementation)
- $A \vee \texttt{true} \equiv \texttt{true}$
- $A \wedge \texttt{false} \equiv \texttt{false}$ (zero laws)
- $A \vee \texttt{false} \equiv A$
- $A \wedge \texttt{true} \equiv A$ (identity laws)

Remark: Sets form a Boolean algebra.

Definition 2.4.2: We write $G[F/H]$ to mean the formula obtained from G by replacing every occurrence of H in G by F.

This is a substitution.

Formally, $G[F/H] \coloneqq F$ if $G = H$. For $F \neq H$, we proceed by structural induction:
- Base case: $x[F/H] \coloneqq x$ for all $x \in X$
- Inductive steps:
- ‣ $(\neg G)[F/H] = \neg(G[F/H])$
- ‣ $(G_1 \wedge G_2)[F/H] = (G_1[F/H]) \wedge (G_2[F/H])$
- ‣ $(G_1 \vee G_2)[F/H] = (G_1[F/H]) \vee (G_2[F/H])$

Theorem 2.4.3 (Substitution theorem): Let F, G, G', H be formulae s.t. $G' = G[F/H]$ and $F \equiv H$, then $G' \equiv G$.

Proof: By structural induction on G.

If $G \equiv H$ then $G[F/H] = F \equiv H$, hence $G' \equiv G$.

Otherwise:
- If $G = x$, then $G' = x$, hence $G \equiv G'$
- If $G = \neg J$, then $G' = G[F/H] = \neg(J[F/H])$, by inductive hypothesis, $= \neg J'$ where $J' \equiv J$
- Conjunction and disjunction follow similarly.

□

Definition 2.4.4: A proof by equational reasoning of $F \equiv G$ is a sequence $F_1, ..., F_n$ such that $F_1 = F$, $F_n = G$, and F_{i+1} is obtained from F_i by a substitution according to the Boolean algebra axioms.

Theorem 2.4.5 (soundness): If we have an equational proof starting in F and ending in G, then $F \equiv G$.

Proof: Consequence of the substitution theorem. □

Definition 2.4.6: A formula is in negation normal form if negation appears only in front of propositional variables.

Lemma 2.4.7: Every formula can be transformed into DNF by equational reasoning.

Proof:

We exhaustively apply de Morgan's laws, and rewrite $\neg$`true` $\equiv$ `false` and $\neg$`false` $\equiv$ `true`. This gives us the negation normal form of F.

Then we exhaustively apply distributivity and the identity laws.

We can then reintroduce any variables x_i that were eliminated, by replacing any disjunct D with $(D \wedge x_i) \vee (D \wedge \neg x_i)$. Thus we obtain a canonical (up to disjunct ordering) DNF for the formula. □

Theorem 2.4.8 (comleteness): If $F \equiv G$, then there is an equational proof starting in F and ending in G.

Proof: Given F, G s.t. $F \equiv G$, then F and G have the same truth table, hence the same DNF H. Then we apply Lemma 2.4.7 to F to obtain H by proof P_1, and to G to obtain H by proof P_2.

Then the equational proof of $F \equiv G$ is P_1 concatenated with the reverse of P_2. □

2.5. Resolution

Use formulae in CNF, presented as sets.

E.g. if $F = \overbrace{(p_1 \vee \neg p_2)}^{\text{clause}} \wedge (p_3 \vee \neg p_4 \vee p_5) \wedge (\neg p_2)$.

We represent this as $\left\{ \overbrace{\{p_1, \neg p_2\}}^{\text{clause}}, \{p_3, \neg p_4, p_5\}, \{\neg p_2\} \right\}$.

We represent the empty clause as $\square \equiv$ `false`. $\{\square\} \equiv$ `false`, $\{\} \equiv$ `true`.

Remark: The set representation naturally expresses commutativity, associativity and idempotence.

Definition 2.5.1: Given literal L, we define

$$\bar{L} = \begin{cases} \neg p & \text{if } L = p \\ p & \text{if } L = \neg p. \end{cases}$$

Definition 2.5.2: Let C_1, C_2 be clauses s.t. $L \in C_1$ and $\bar{L} \in C_2$. Then the resolvent of C_1 and C_2 is

$$R = (C_1 \setminus \{L\}) \cup \left(C_2 \setminus \left\{\bar{L}\right\}\right).$$

We say that R is derived by resolution from C_1 and C_2; we write $\dfrac{C_1 \quad C_2}{\mathrm{R}}$.

Definition 2.5.3: A derivation (or proof) of a clause C from the set of clauses F is a sequence $C_1, ..., C_m$ of clauses such that

- $C_m = C$
- For each $1 \leq i \leq m$, either $C_i \in F$ (an assumption), or C_i is the resolvent of C_j and C_k, for some $j \neq k, j, k < i$.

Definition 2.5.4: A derivation of $\square$ demonstrates a refutation of F.

Remark: To show that $\mathcal{S} \models F$, we can demonstrate a refutation of $\mathcal{S} \cup \{\neg F\}$.

Definition 2.5.5: Given F, define

$$\mathrm{Res}(F) = F \cup \{R \mid R \text{ is a resolvent from two clauses in } F\}.$$

Also define $\mathrm{Res}^0(F) = F$; $\mathrm{Res}^{n_1}(F) = \mathrm{Res}(\mathrm{Res}^n(F))$.

Also $\mathrm{Res}^*(F) = \bigcup_{n>0} \mathrm{Res}^n(F)$. Note that this is computable in finite time. TODO is it?

Proposition 2.5.6: $C \in \mathrm{Res}^*(F)$ iff there exists a resolution derivation of C from F.

Lemma 2.5.7 (Resolution lemma): If R is the resolvent of $C_1, C_2 \in F$, then $F \equiv F \cup \{R\}$.

Proof: $F \equiv F \cup \{R\}$ if we have that $\mathcal{A} \vDash F$ iff $\mathcal{A} \vDash F \cup \{R\}$.

$\Longleftarrow$: If $\mathcal{A} \vDash F \cup \{R\}$, then clearly $\mathcal{A} \vDash F$.

$\Longrightarrow$: Let $\mathcal{A} \vDash F$, $R = \{C_1 \setminus \{L\}\} \cup \left\{C_2 \setminus \left\{\bar{L}\right\}\right\}$. Then we have two cases:
1. $\mathcal{A} \vDash L$, then since $\mathcal{A} \vDash C_2$, we have $\mathcal{A} \vDash C_2 \setminus \left\{\bar{L}\right\}$. Hence $\mathcal{A} \vDash R$.
2. $\mathcal{A} \vDash \neg L$, then since $\mathcal{A} \vDash C_1$, we have $\mathcal{A} \vDash C_1 \setminus \{L\}$, so $\mathcal{A} \vDash R$.

□

Theorem 2.5.8 (soundness): If $\square$ can be derived by resolution from F, then F is unsatisfiable.

Proof: By induction on the length of the resolution proof.

$F \equiv F_1 \equiv ... \equiv F_n \equiv \square \equiv \texttt{false}$. □

Theorem 2.5.9 (completeness): If F is unsatisfiable, then $\square$ can be derived from F by resolution.

Proof: By induction on the number of propositional variables that appear in F, n.

If $n = 0$, then $F = \{\square\}$. Trivial.

Then suppose true for n, and consider $n + 1$.

F mentions prop. vars. $p_1, ..., p_{n+1}$. Let $F_0 = F[\texttt{false}\ /p_{n+1}], F_1 = F[\texttt{true}\ /p_{n+1}]$.

Since F is unsatisfiable, both F_0 and F_1 are unsatisfiable. By the inductive hypothesis, we have $C_0, C_1, ..., C_m = \square$ being a refutation of F_0.

Note that C_i or $C_i \cup \{p_{n+1}\}$ are already in F. By reintroducing p_{n+1}, we either obtain a proof of $\square$, or of p_{n+1}.

In the latter case, we can apply the same reasoning to F_1 and obtain a proof of $\neg p_{n+1}$. Then the final resolution step gives us $\square$. □

Remark: Constructing equivalent CNF formulae can be expensive. But because resolution only checks unsatisfiability, we only need an equisatisfiable formula.

Give F, do the following:
- Introduce fresh prop. vars for every subformula G of F, whenever G is not a literal; call it x_G
- Introduce $x_G \leftrightarrow G'$, where G' is G, with the top-level subformulae replaced by the new prop. vars.
- Use equational transformation to transform all of the $x_G \leftrightarrow G'$s into CNF

- The final formula is all new CNF formulae, plus x_F.

Example: Consider the formula $F = \neg(p \wedge q) \wedge r$. The subformulae of F, excluding literals, are $\{\neg(p \wedge q) \wedge r, \neg(p \wedge q), p \wedge q\}$. Then we introduce new propositional variables x_F, $x_{\neg(p\wedge q)}$, $x_{p\wedge q}$, and say that:

- $x_F \leftrightarrow x_{\neg(p\wedge q)} \wedge r$
- $x_{\neg(p\wedge q)} \leftrightarrow \neg x_{p\wedge q}$
- $x_{p\wedge q} \leftrightarrow p \wedge q$

We then transform these into CNF:

- $$\begin{aligned}&\left(\neg x_F \vee \left(x_{\neg(p\wedge q)} \wedge r\right)\right) \wedge \left(\neg\left(x_{\neg(p\wedge q)} \wedge r\right) \vee x_F\right)\\ &\quad\equiv \left(\neg x_F \vee x_{\neg(p\wedge q)}\right) \wedge (\neg x_F \vee r) \wedge \left(\neg x_{\neg(p\wedge q)} \vee \neg r \vee x_F\right)\end{aligned}$$
- $$\left(x_{\neg(p\wedge q)} \to \neg x_{p\wedge q}\right) \wedge \left(\neg x_{p\wedge q} \to x_{\neg(p\wedge q)}\right) \equiv \left(\neg x_{\neg(p\wedge q)} \vee \neg x_{p\wedge q}\right) \wedge \left(x_{p\wedge q} \vee x_{\neg(p\wedge q)}\right)$$
- $$\begin{aligned}\left(x_{p\wedge q} \to (p \wedge q)\right) \wedge \left((p \wedge q) \to x_{p\wedge q}\right) &\equiv \left(\neg x_{p\wedge q} \vee (p \wedge q)\right) \wedge \left(\neg(p \wedge q) \vee x_{p\wedge q}\right)\\ &\equiv \left(\neg x_{p\wedge q} \vee p\right) \wedge \left(\neg x_{p\wedge q} \vee q\right) \wedge \left(\neg p \vee \neg q \vee x_{p\wedge q}\right)\end{aligned}$$

  and combine them all with x_F to get:

$$\begin{aligned}&\{\\ &\quad\{x_F\},\\ &\quad\{\neg x_{p\wedge q}, p\}, \{\neg x_{p\wedge q}, q\}, \{\neg p, \neg q, x_{p\wedge q}\},\\ &\quad\left\{\neg x_{\neg(p\wedge q)}, \neg x_{p\wedge q}\right\}, \left\{x_{p\wedge q}, x_{\neg p\wedge q}\right\},\\ &\quad\left\{\neg x_f, x_{\neg(p\wedge q)}\right\}, \{\neg x_F, r\}, \left\{\neg x_{\neg(p\wedge q)}, \neg r, x_F\right\}\\ &\}.\end{aligned}$$

2.6. Natural deduction

Definition 2.6.1: The calculus of natural deduction has no axioms. Proofs begin by assumptions, and we use rules for Boolean connectives. A proof is a tree.

Temporary assumptions that are discharged are denoted by square brackets in the proof rules. If, at the end of a proof, all assumptions are discharged, the proof is valid.

Definition 2.6.2: The natural deduction rules are as follows:

- Conjunction introduction:

$$\frac{A \quad B}{A \wedge B} \wedge\text{I}$$

- Conjunction elimination:

$$\frac{A \wedge B}{A} \wedge\text{E} \qquad \frac{A \wedge B}{B} \wedge\text{E}$$

- Disjunction introduction:

$$\frac{A}{A \vee B} \vee\text{I} \qquad \frac{B}{A \vee B} \vee\text{I}$$

- Disjunction elimination:

$$\frac{\begin{array}{ccc} & [A] & [B] \\ & \vdots & \vdots \\ A \vee B & C & C \end{array}}{C} \vee\text{E}$$

  This says that, in order to derive C from $A \vee B$, it is sufficient to derive C from A and also C from B. Then both assumptions $[A]$ and $[B]$ are discharged. This is informally proof by cases.

- Implication introduction:

$$\frac{\begin{array}{c}[A] \\ \vdots \\ B\end{array}}{A \to B} \to\text{I}$$

- Implication elimination:

$$\frac{A \to B \quad A}{B} \to\text{E}$$

- Negation introduction:

$$\frac{\begin{array}{c}[A] \\ \vdots \\ \bot\end{array}}{\neg A} \neg\text{E}$$

- Negation elimination:

$$\frac{\neg A \quad A}{\bot} \neg\text{E}$$

- Ex falso quodlibet:

$$\frac{\bot}{D} \bot\text{J}$$

- Reductio ad absurdum:

$$\dfrac{\begin{array}{c}[\neg A]\\ \vdots \\ \bot\end{array}}{A}\,\bot\mathrm{K}$$

Remark: Without $\bot$J or $\bot$K, natural deduction is equivalent to $\mathbf{M}_0$; with $\bot$J it is equivalent to $\mathbf{J}_0$; and with $\bot$K, it is equivalent to $\mathbf{K}_0$.

Definition 2.6.3: A deduction of a formula F is a finite tree of formulae in which every leaf is an assumption, and every other formula is the conclusion of an application of one of the inference rules. The open assumptions of a deduction are those that are not discharged by any rule in the tree. A deduction of F with no open assumptions is a proof of F, and F is a theorem if such a proof exists.

Remark: Natural deduction is sound and complete.

2.7. Sequent calculus

Definition 2.7.1: A sequent is an expression of the form

$$A_1, ..., A_n \Longrightarrow B_1, ..., B_m,$$

where the A_is and B_is are propositional forumlae.

Such a sequent is valid if the following is valid in classical logic:

- $A_1 \wedge ... \wedge A_n \to B_1 \vee ... \vee B_m$, if $n, m \neq 0$
- $B_1 \vee ... \vee B_m$, if $n = 0, m \neq 0$
- $\neg(A_1 \wedge ... \wedge A_n)$, if $n \neq 0, m = 0$,
- false otherwise.

The left-hand side of the sequent is the antecedent, and the right-hand side is the succedent.

Remark: The sequent with an empty antecedent and empty succedent is the empty sequent and is always invalid, by the last case in the definition above.

Remark: $A \Longrightarrow A$ is a tautology/axiom.

Remark: In declarations of sequent inference rules, we use the Greek letters $\Gamma, \Delta, \Sigma, \Pi, \Phi, \Theta, \Lambda, \Xi$ to denote (possibly empty) ordered lists of propositional formulae.

Definition 2.7.2: The inference rules of sequent calculus are either structural rules, which manipulate the list structure of a sequent, and operational rules, which provide rules for introducing logical connectives in the antecent or succedent. Inference rules take one or more premises from which a conclusion is drawn.

Definition 2.7.3: The structural rules are as follows:

- Interchange allows consecutive formulae to be swapped, on both the left and right-hand sides:

$$\frac{\Delta, A, B, \Gamma \Rightarrow \Theta}{\Delta, B, A, \Gamma \Rightarrow \Theta}\,\text{IL} \qquad \frac{\Gamma \Rightarrow \Theta, A, B, \Lambda}{\Gamma \Rightarrow \Theta, B, A, \Lambda}\,\text{IR}$$

- Weakening allows formulae to be added to the left or right:

$$\frac{\Gamma \Rightarrow \Theta}{\Gamma, A \Rightarrow \Theta}\,\text{WL} \qquad \frac{\Gamma \Rightarrow \Theta}{\Gamma \Rightarrow \Theta, A}\,\text{WR}$$

- Contraction allows duplicate formulae to be merged:

$$\frac{A, A, \Gamma \Rightarrow \Theta}{A, \Gamma \Rightarrow \Theta}\,\text{CL} \qquad \frac{\Gamma \Rightarrow \Theta, A, A}{\Gamma \Rightarrow \Theta, A}\,\text{CR}$$

Definition 2.7.4: The operational rules are as follows:

- Conjunction:

$$\frac{A, \Gamma \Rightarrow \Theta}{A \wedge B, \Gamma \Rightarrow \Theta}\,\wedge\text{L} \qquad \frac{B, \Gamma \Rightarrow \Theta}{A \wedge B, \Gamma \Rightarrow \Theta}\,\wedge\text{L} \qquad \frac{\Gamma \Rightarrow \Theta, A \quad \Gamma \Rightarrow \Theta, B}{\Gamma \Rightarrow \Theta, A \wedge B}\,\wedge\text{R}$$

- Disjunction:

$$\frac{A, \Gamma \Rightarrow \Theta \quad B, \Gamma \Rightarrow \Theta}{A \vee B, \Gamma \Rightarrow \Theta}\,\vee\text{L} \qquad \frac{\Gamma \Rightarrow \Theta, A}{\Gamma \Rightarrow \Theta, A \vee B}\,\vee\text{R} \qquad \frac{\Gamma \Rightarrow \Theta, B}{\Gamma \Rightarrow \Theta, A \vee B}\,\vee\text{R}$$

- Conditional:

$$\frac{\Gamma \Rightarrow \Theta, A \quad B, \Delta \Rightarrow \Lambda}{A \to B, \Gamma, \Delta \Rightarrow \Theta, \Lambda}\,\to\text{L} \qquad \frac{A, \Gamma \Rightarrow \Theta, B}{\Gamma \Rightarrow \Theta, A \to B}\,\to\text{R}$$

- Negation:

$$\frac{\Gamma \Rightarrow \Theta, A}{\neg A, \Gamma \Rightarrow \Theta}\,\neg\text{L} \qquad \frac{A, \Gamma \Rightarrow \Theta}{\Gamma \Rightarrow \Theta, \neg A}\,\neg\text{R}$$

Definition 2.7.5: A proof in the sequent calculus is a finite tree of sequents in which every leaf is an axiom ($A \Rightarrow A$), and every other node is obtained from its children (where children are written above their parent) by an inference rule.

A sequent is provable or derivable if it is the root of such a proof.

Remark: To prove validity of F in the sequent calculus, we want to obtain an empty left-hand side, with F on the right:

$$\frac{\vdots}{\Longrightarrow F}.$$

Example: To prove that $(p \wedge q) \to p$:

$$\dfrac{\dfrac{\dfrac{\dfrac{p \Rightarrow p}{q, p \Rightarrow p}\,\mathrm{WL}}{p, q \Rightarrow p}\,\mathrm{IL}}{p \wedge q \Longrightarrow p}\,\wedge L}{\Longrightarrow (p \wedge q) \to p}\,\to R$$

Example: To prove $(p \to q) \to (\neg q \to \neg p)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p \Longrightarrow p}{\Longrightarrow p, \neg p}\,\neg\mathrm{R} \qquad q \Longrightarrow q}{p \to q \Longrightarrow \neg p, q}\,{\to}\mathrm{L}}{\neg q, p \to q \Longrightarrow \neg p}\,\neg\mathrm{L}}{p \to q \Longrightarrow \neg q \to \neg p}\,{\to}\mathrm{R}}{\Longrightarrow (p \to q) \to (\neg q \to \neg p))}\,{\to}\mathrm{R}$$

Theorem 2.7.6: The sequent calculus is a sound and complete proof system for classical logic.

2.8. The compactness theorem

Theorem 2.8.1 (compactness theorem):

A set of formulae $\mathcal{S}$ is satisfiable iff every finite subset of $\mathcal{S}$ is satisfiable.

Equivalently, a set of formulae $\mathcal{S}$ is unsatisfiable iff there is some unsatisfiable finite $\mathcal{S}' \subseteq \mathcal{S}$.

Proof:

"$\Longrightarrow$": Suppose that $\mathcal{S}$ is satisfiable, then there exists $\mathcal{A}$ s.t. $\mathcal{A} \vDash F$ for all $F \in \mathcal{S}$, hence $\mathcal{A} \vDash G$ for all $G \in \mathcal{S}'$ for any finite subset $\mathcal{S}' \subseteq S$.

"$\Longleftarrow$":

Define a partial assignment to be a function $\mathcal{A} : X \to \{0, 1\}$, where $\mathrm{dom}(\mathcal{A}) = \{x_1, ..., x_n\}$.

We say that a partial assignment $\mathcal{A}$ is good if $\mathcal{A} \vDash F$ for all $F \in \mathcal{S}$ s.t. F only mentions the variables in $\mathrm{dom}(\mathcal{A})$.

Suppose that every finite subset of $\mathcal{S}$ is satisfiable, then for every n, there is a partial assignment $\mathcal{A}$ s.t. $\mathrm{dom}(\mathcal{A}) = \{x_1, ..., x_n\}$ that is good:

- Let $\mathcal{S}' \in S$ be all formulae over $x_1, ..., x_n$.
- $\mathcal{S}'$ might be infinite, but contains only finitely many formulae, up to logical equivalence.
- Choose representatives for each equivalence class and put them in $\mathcal{S}'' \subseteq \mathcal{S}'$, $\mathcal{S}''$ finite.
- By assumption, there is some $\mathcal{A}$ such that $\mathcal{A} \vDash F$ for all $F \in \mathcal{S}''$. Moreover, $\mathcal{A}$ will satisfy every formula in $\mathcal{S}'$, because they are all logically equivalent to a formula in $\mathcal{S}''$

Now we plan to construct a sequence $\mathcal{A}_0, \mathcal{A}_1, ...$ of partial assignments that are good, and such that $\mathcal{A}_{i+1}$ extends $\mathcal{A}_i$; that is, $\mathrm{dom}(\mathcal{A}_i) \subseteq \mathrm{dom}(\mathcal{A}_{i+1})$, and $\mathcal{A}_i(x_j) = \mathcal{A}_{i+1}(x_j)$ for all $x_j \in \mathrm{dom}(\mathcal{A}_i)$.

We do this while maintaining the invariant that, for all $n \geq 0$, there are infinitely many good extensions of $\mathcal{A}_n$.

We construct the $\mathcal{A}_n$s by induction on n.

The base case is clear: $\mathcal{A}_0$ is the empty assignment, which vacuously satisfies the empty subset of $\mathcal{S}$, and there are clearly an infinite number of good assignments that extend the empty assignment, since every finite subset of $\mathcal{S}$ is satisfiable.

Then suppose we constructed $\mathcal{A}_1, ..., \mathcal{A}_n$, and consider two assignments $\mathcal{B}, \mathcal{B}'$ that extend $\mathcal{A}_n$, with $\mathcal{B}(x_{n+1}) = 0, \mathcal{B}(x_{n+1}) = 1$. Since $\mathcal{A}_n$ has infinitely many good extensions, one of $\mathcal{B}$ and $\mathcal{B}'$ must have infinitely many good extensions. Pick the $\mathcal{B}$ or $\mathcal{B}'$ that has infinitely many good extensions to be $\mathcal{A}_{n+1}$. Then by induction we have constructed our sequence $\mathcal{A}_0, ..., \mathcal{A}_n$.

Now we take $\mathcal{A}$ such that $\mathcal{A}(x_i) = \mathcal{A}_i(x_i)$ for all $i \geq 1$. Then $\mathcal{A} \vDash F$ for all $F \in \mathcal{S}$, because we can take the largest index of a prop. var. that appears in F, and we know that $\mathcal{A}$ extends a model of F, so $\mathcal{A} \vDash F$.

□

Example (an application of the compactness theorem):

Proposition: Let $G := (V, E)$ be a graph with $V = \{v_i \mid i \in \mathbb{N}\}$, and suppose that every finite subgraph of G is k-colourable, then G is k-colourable.

Proof: For every $v \in V$, introduce a propositional variabel $x_{v,i}$ (v has colour i).

$$F_v := \bigvee_{i=1}^{k} x_{v,i} \qquad \forall v \in V$$

$$G_v := \bigwedge_{i=1}^{k-1} \bigwedge_{j=i+1}^{k} \neg(x_{v,i} \wedge x_{v,j}) \qquad \forall v \in V$$

$$H_{u,v} := \bigwedge_{i=1}^{k} \neg(x_{v,i} \wedge x_{u,i}) \qquad \forall (u,v) \in E$$

Then let

$$\mathcal{S} := \{F_v, G_v \mid v \in V\} \cup \{H_{u,v} \mid (u,v) \in E\},$$

and we claim that $\mathcal{S}$ is satisfiable iff G has a colouring (proof is easy).

Observe that any finite $S' \subseteq S$ "talks about" a finite subgraph of G; by assumption, S' is satisfiable. By the compactness theorem, S is satisfiable; therefore G is k-colourable. □

3. 1st-order logic

3.1. Syntax

Definition 3.1.1: A signature, σ, is a tuple of
- constant symbols (c, d)
- function symbols (f, g)
- predicate symbols (P, Q, R)

with function and predicate symbols each having nonzero arity k.

We also keep a set of variables $X = \{x_1, x_2, ...\}$, independently from the signature.

Definition 3.1.2: A σ-term is defined by structural induction:
- every x is a term
- every c is a term
- if $t_1, ..., t_k$ are terms, then $f(t_1, ..., t_k)$ is a term (assuming f has arity k)
- nothing else is a term

Definition 3.1.3: A formula of 1st-order logic over σ is defined by structural induction.

- $P(t_1, ..., t_k)$ is an atomic formula for a k-ary relation symbol P and terms $t_1, ..., t_k$

- If F, G are formulae, and $x \in X$, then the following are formulae:
- ▸ $\neg F$, $F \wedge G$, $F \vee G$
- ▸ $\exists x F$ (existential quantifier)
- ▸ $\forall x F$ (universal quantifier)
- Nothing else is a formula

Definition 3.1.4: For existential/universal quanitifiers $\exists x F$ / $\forall x F$, we say that F is in the scope of $\exists x$ / $\forall x$, and moreover x is bound by $\exists x$ / $\forall x$.

If a variable is not bound, it is free.

A formula with no free variables is a sentence, or closed.

TODO quantifier depth

3.2. Semantics

Definition 3.2.1: A σ-structure or σ-assignment $\mathcal{A}$ is a tuple consisting of:
- A non-empty universe $\mathcal{U}_\mathcal{A}$
- For every k-ary function symbol f, a function $f_\mathcal{A} : \mathcal{U}_\mathcal{A}^k \to \mathcal{U}_\mathcal{A}$
- For every k-ary predicate symbol P, a k-ary relation $P_\mathcal{A} \subseteq \mathcal{U}_\mathcal{A}^k$
- For every constant c, a $c_\mathcal{A} \in \mathcal{U}_\mathcal{A}$
- For every variable x, a $x_\mathcal{A} \in \mathcal{U}_\mathcal{A}$.

Definition 3.2.2: For σ-structure $\mathcal{A}$ and term t, we define the value of t under $\mathcal{A}$ by structural induction:
- $\mathcal{A}(c) \coloneqq c_\mathcal{A} \in \mathcal{U}_\mathcal{A}$
- $\mathcal{A}(x) \coloneqq x_\mathcal{A} \in \mathcal{U}_\mathcal{A}$
- $\mathcal{A}(f(t_1, ..., t_k)) \coloneqq f_\mathcal{A}(\mathcal{A}(t_1), ..., \mathcal{A}(t_k))$

Definition 3.2.3: We define the satisfaction relation $\mathcal{A} \vDash F$ by structural induction:
- $\mathcal{A} \vDash P(t_1, ..., t_k)$ iff $(\mathcal{A}(t_1), ...\mathcal{A}(t_k)) \in P_\mathcal{A}$
- $\mathcal{A} \vDash \neg F$ iff $\mathcal{A} \nvDash F$
- $\mathcal{A} \vDash F \wedge G$ iff $\mathcal{A} \vDash F$ and $\mathcal{A} \vDash G$
- $\mathcal{A} \vDash F \vee G$ iff $\mathcal{A} \vDash F$ or $\mathcal{A} \vDash G$
- $\mathcal{A} \vDash \exists x F$ iff there is $a \in \mathcal{U}_\mathcal{A}$ s.t. $\mathcal{A}_{[x \mapsto a]} \vDash F$
- $\mathcal{A} \vDash \forall x F$ iff for all $a \in \mathcal{U}_\mathcal{A}$, $\mathcal{A}_{[x \mapsto a]} \vDash F$

Definition 3.2.4: A 1st-order logic formula F is satisfiable if there is a σ-structure $\mathcal{A}$ s.t. $\mathcal{A} \vDash F$.

F is unsatisfiable if there is no such structure.

Definition 3.2.5: 1st-order logic formulae F, G are equivalent ($F \equiv G$) if $\mathcal{A} \vDash F \Leftrightarrow \mathcal{A} \vDash G$ for all $\mathcal{A}$.

Definition 3.2.6: 1st-order formulae are equisatisfiable if F is satisfiable iff G is satisfiable.

3.3. Equivalences & Skolem form

Definition 3.3.1: A formula G is in Skolem form if $G = \forall x_1 \forall x_2 ... \forall x_k F$, where F is quantifier-free.

Lemma 3.3.2 (Relevance Lemma): Let F be a formula, and assignments $\mathcal{A}, \mathcal{A}'$ coinciding on their interpretation of constants, function symbols, predicate symbols and variables that are free in F. Then $\mathcal{A} \vDash F$ iff $\mathcal{A}' \vDash F$.

Remark: Propositional logic laws for logical equivalence still carry over, e.g. $(\neg F \wedge G) \equiv \neg F \vee \neg G$.

Moreover, logical equivalence is still a congruence, e.g. $F \wedge G \equiv F' \wedge G'$ iff $F \equiv F'$ and $G \equiv G'$.

Also, if $F \equiv G$, then $\forall x F \equiv \forall x G$, and $\exists x F \equiv \exists x G$.

Theorem 3.3.3:

Let F, G be formulae, then the following equivalences hold in 1st-order logic:

(a) $\neg \forall x F \equiv \exists x \neg F$, $\neg \exists x F \equiv \forall x \neg F$ (these allow us to establish a negation-normal form for 1st-order logic)

(b) If x does not appear free in G, then
- $\forall x F \wedge G \equiv \forall x (F \wedge G)$, $\forall x F \vee G \equiv \forall x (F \vee G)$
- $\exists x F \wedge G \equiv \exists x (F \wedge G)$, $\exists x F \vee G \equiv \exists x (F \vee G)$

(c) $\forall x F \wedge \forall x G \equiv \forall x (F \wedge G)$, $(\exists x F \vee \exists x G) \equiv \exists x (F \vee G)$

(d) $\forall x \forall y F \equiv \forall y \forall x F$, $\exists x \exists y F \equiv \exists y \exists x F$

Proof:

(a) $A \vDash \neg \forall x F$ iff $\mathcal{A} \nvDash \forall x F$

iff $\mathcal{A}_{[x \mapsto a]} \nvDash F$ for some $a \in \mathcal{U}_{\mathcal{A}}$

iff $\mathcal{A}_{[x \mapsto a]} \vDash \neg F$ for some $a \in \mathcal{U}_{\mathcal{A}}$

iff $\mathcal{A} \vDash \exists x \neg F$

and similar for the second equivalence.

(b) $\mathcal{A} \vDash (\forall x F) \wedge G$ iff $\mathcal{A} \vDash \forall x F$ and $\mathcal{A} \vDash G$

iff, for all $a \in \mathcal{U}_{\mathcal{A}}$, $\mathcal{A}_{[x \mapsto a]} \vDash F$ and $\mathcal{A} \vDash G$

iff, for all $a \in \mathcal{U}_{\mathcal{A}}$, $\mathcal{A}_{[x \mapsto a]} \vDash F$ and $\mathcal{A}_{[x \mapsto a]} \vDash G$

(by the Relevance Lemma, since x does not appear free in G)

iff, for all $a \in \mathcal{U}_{\mathcal{A}}$, $\mathcal{A}_{[x \mapsto a]} \vDash F \wedge G$

iff $\mathcal{A} \vDash \forall x (F \wedge G)$

and similarly for the other equivalences.

(c) TODO exercise

(d) TODO exercise

□

Definition 3.3.4: A formula G is in prenex form if

$$G = Q_1 x_1 ... Q_n y_n G', \; Q_i \in \{\exists, \forall\},$$

where G' is quantifier-free.

Lemma 3.3.5: Any formula F is equivalent to one in prenex form.

Definition 3.3.6: Let F be a formula, x a variable, and t a term, then $F[t/x]$ is a substitution obtained by replacing every free occurrence of x in F with t.

TODO define by structural induction.

Lemma 3.3.7 (translation-lemma): If t is a term and F a formula such that no variable in t occurs bound in F, then we have that $\mathcal{A} \vDash F[t/x]$ iff $\mathcal{A}_{[x \mapsto \mathcal{A}(t)]} \vDash F$.

Proof: TODO, optional □

Lemma 3.3.8: Let $F = Q\, x\, G$ with Q a quantifier, and y a variable not occurring in G. Then $F \equiv Q\, y\, G[y/x]$.

Proof: TODO □

Definition 3.3.9: A formula is rectified if no variable occurs both bound and free at the same time, and all quantifiers refer to different variables.

Lemma 3.3.10: Every formula is equivalent to a rectified formula, and moreover one in rectified prenex form.

Proof: TODO by the lemmas above. □

Lemma 3.3.11: Let $F = \forall y_1 \forall y_2 ... \forall y_k \exists z G$ where G is rectified. Let f be a fresh function symbol of arity k, then F is equisatisfiable with $F' := \forall y_1 \forall y_2 ... \forall y_k G[f(y_1, ..., y_k)/z]$.

Proof sketch: Suppose $\mathcal{A} \vDash F$. We define $\mathcal{A}'$ extending $\mathcal{A}$ such that $f_{\mathcal{A}'}(a_1, ..., a_k) := a$, where a is such that $\mathcal{A}_{[y_1 \mapsto a_1]...[y_k \mapsto a_k][x \mapsto a]} \vDash G$.

Then $\mathcal{A}' \vDash F'$. □

Remark: If $k = 0$, then f is a constant symbol.

Theorem 3.3.12: Any 1st-order formula F is equisatisfiable with a formula in Skolem form.

Proof: We get F in prenex form, and utilise Lemma 3.3.11 to eliminate existential quanitifiers from left to right. □

Remark: The quantifier-free part of a Skolem-form formula is sometimes known as the matrix of the formula.

3.4. Herbrand's Theorem & Ground Resolution

Definition 3.4.1: A ground term (of σ) is a variable-free term.

Definition 3.4.2: A σ-structure $\mathcal{H}$ is a Herband structure if the following are true:

- $\mathcal{U}_{\mathcal{H}}$ is the set of ground terms
- $c_{\mathcal{H}} = c$ for every constant symbol c
- For every k-ary function symbol f and ground terms $t_1, ..., t_k$, $f_{\mathcal{H}}(t_1, ..., t_k) = f(t_1, ..., t_k)$

Remark: The interpretations of constant and function symbols are just strings of symbols.

Remark: The only thing we are free to choose is the interpretation of predicate symbols.

Remark: The interpretation of a term in a Herbrand structure is $\mathcal{H}(t) = t$.

Corollary 3.4.3 (translation lemma for Herbrand structures):

$\mathcal{H} \vDash F[t/x]$ iff $\mathcal{H}_{[x \mapsto t]} \vDash F$.

Theorem 3.4.4 (Herbrand's theorem): Let $F = \forall x_1 ... x_k F^*$ be a **closed** formula in Skolem form. Then F is satisfiable iff F has a Herbrand model.

Proof:

"$\Longleftarrow$": obvious.

"$\Longrightarrow$": Suppose $\mathcal{A} \vDash F$. We define $(t_1, ..., t_n) \in P_{\mathcal{H}}$ iff $\mathcal{A} \vDash P(t_1, ..., t_n)$. Then we show that $\mathcal{H}$ is a model of F by induction on the number of quantifiers k.

If $k = 0$, then F is a Boolean combination of atomic formulae $P(t_1, ..., t_n)$ for ground terms $t_1, ..., t_n$. By construction, $\mathcal{H} \vDash P(t_1, ..., t_k)$ iff $\mathcal{A} \vDash P(t_1, ..., t_k)$, so $\mathcal{H} \vDash F$, iff $\mathcal{A} \vDash F$.

Then suppose true for Skolem-form sentences with k quantifiers, and consider F with $k + 1$ quantifiers, where $F = \forall x_1 G$. By the translation lemma, $\mathcal{A} \vDash G[t/x_1]$ iff $\mathcal{A}_{[x_1 \mapsto \mathcal{A}(t)]} \vDash G$. Hence $\mathcal{A} \vDash G[t/x_1]$ for all ground terms t. $G[t/x_1]$ is closed, so by the i.h. $\mathcal{H} \vDash G[t/x_1]$ for all ground terms t. By the translation lemma, $\mathcal{A}_{[x_1 \mapsto t]} \vDash G$ for all $t \in \mathcal{U}_{\mathcal{H}}$, hence $\mathcal{H} \vDash \forall x_1 G$, . □

Corollary 3.4.5: For any formula with an uncountable model, there is a countable model.

Definition 3.4.6: Let $F = \forall x_1 ... \forall x_n F^*$ be closed in Skolem form, and define the Herbrand expansion $E(F)$ to be

$$E(F) := \{F^*[t_1/x_1]....[t_n/x_n] \mid t_1, ...t_n \text{ are ground terms}\}.$$

Remark: Each formula in $E(F)$ is a Boolean combination of atomic formulae. This means that $E(F)$ has a Herbrand model iff it is propositionally satisfiable, i.e. there is an assignment to the set of closed atomic formulae that makes all formulae in $E(F)$ evaluate to true.

Theorem 3.4.7: A closed Skolem-form formula $F := \forall x_1 ... \forall x_k F^*$ is satisfiable iff $E(F)$ is satisfiable when viewing atomic formulae as propositional variables.

Proof:

F sat. iff exists a Herbrand model $\mathcal{H}$ of F, by Herbrand's Theorem
iff $\mathcal{H} \models F$
iff $\mathcal{H}_{([x_1 \mapsto t_1]...[x_n \mapsto t)n]}) \models F^*$ for all ground terms $t_1, ..., t_n \in \mathcal{U}_\mathcal{H}$
iff $\mathcal{H} \models F^*[t_1/x_1]...[t_n/x_n]$, by the Translation Lemma for Herbrand structures
iff $\mathcal{H} \models E(F)$ by defn of Herbrand expansion
iff $E(F)$ is satisfiable, by Herbrand's Theorem. □

Theorem 3.4.8 (Ground Resolution): A closed Skolem-form formula F is unsatisfiable iff there is a propositional resolution refutation of $E(F)$.

Proof: TODO □

Remark:

To prove validity of F, we can provide a ground resolution refutation of $G := \neg F$.

- Transform into negation normal form
- Transform into rectified prenex form
- Skolemise
- Transform into CNF
- Do a propositional resolution proof on the Herband expansion
- If $\square$ is derived, then G is unsatisfiable, so F is valid

Example:

Let $F = (\forall x(P(x) \to Q(x)) \wedge \exists x P(x)) \to \exists x Q(x)$, $G := \neg F$.

$$
\begin{aligned}
G &\equiv \forall x(P(x) \to Q(x)) \land \exists x P(x) \land \neg\exists x Q(x) \\
&\equiv \forall x(\neg P(x) \land Q(x)) \land \exists x P(x) \land \forall x \neg Q(x) \\
&\equiv \forall x(\neg P(x) \lor Q(x)) \land \exists y P(y) \land \forall z \neg Q(z) \\
&\equiv \forall x \exists y \forall z((\neg P(x) \lor Q(x)) \land P(y) \land \neg Q(z)) \\
&\text{equisat w/ } \forall x \forall z((\neg P(x) \lor Q(x)) \land P(f(x)) \land \neg Q(z)) =: H
\end{aligned}
$$

Note that the set of ground terms is empty because there are no constant symbols. However, we can introduce a fresh constant symbol a, and now the set of ground terms is $\{a, f(a), f(f(a)), ...\}$.

$$E(H) = \{\neg P(A) \lor Q(A), P(f(a)), \neg Q(a), \neg P(f(a)) \lor Q(f(a)), \neg Q(f(a)), ...\}$$

TODO resolution refutation

3.5. Natural deduction and sequent calculus in 1st-order logic

Definition 3.5.1: We introduce new natural deduction proof rules for quantifiers:

- Universal elimination:

$$\frac{\forall x A(x)}{A[t/x]} \forall \text{E}$$

- Universal introduction:

$$\frac{A(c)}{\forall x A[x/c]} \forall \text{I}$$

  where c is free and must not occur in any undischarged assumption on which the deduction of $A(c)$ relies.

- Existential introduction:

$$\frac{A[t/c]}{\exists x A[x/c]} \exists \text{I}$$

- Existential elimination:

$$\frac{\exists x A(x) \qquad \begin{array}{c}[A(x)] \\ \vdots \\ C\end{array}}{C} \exists \text{E}$$

  provided that c does not occur in C, and all assumptions $[A(c)]$ must be discharged when $\exists$E is applied.

Example:

$$\dfrac{\dfrac{[\forall x(P(x) \to Q(x)) \land \exists x P(x)]^1}{\exists x P(x)} \land\text{E} \qquad \dfrac{\dfrac{\dfrac{[\forall x(P(x) \to Q(x)) \land \exists x P(x)]^1}{\forall x(P(x) \land Q(x))} \forall\text{E}}{P(x) \to Q(c)} \qquad \dfrac{[P(c)]^2}{} \to\text{E}}{\dfrac{Q(c)}{\exists x Q(x)} \exists\text{I}}}{1 \dfrac{\exists x Q(x)}{(\forall x(P(x) \to Q(x)) \land \exists x P(X)) \to \exists x Q(x)} \to\text{I}}$$

Definition 3.5.2: We extend the sequent calculus with new proof rules for quantifier.

- Universal quantification:

$$\dfrac{A(t), \Gamma \Longrightarrow \Theta}{\forall x A(x), \Gamma \Longrightarrow \Theta} \forall\text{L} \qquad !\dfrac{\Gamma \Longrightarrow \Theta, A(a)}{\Gamma \Longrightarrow \Theta, \forall x A(x)} \forall\text{R}$$

  where the exclamation mark (!) indicates that a must not occur in Γ, Θ.

  - Existential quantification:

$$!\dfrac{A(a), P \Longrightarrow \Theta}{\exists x A(x), \Gamma \Longrightarrow \Theta} \exists\text{L} \qquad \dfrac{\Gamma \Longrightarrow \Theta, A(t)}{\Gamma \Longrightarrow \Theta, \exists x A(x)} \exists\text{R}$$

Example:

$$\dfrac{\dfrac{\dfrac{P(a) \Longrightarrow P(a)}{P(a) \Longrightarrow \exists x P(x)} \exists R}{\forall x P(x) \Longrightarrow \exists x P(x)} \forall\text{L}}{\Longrightarrow \forall x P(x) \to \exists x P(x)} \to\text{R}$$

TODO IMPORTANT: exercise 22 (ground resolution), 23 (natural deduction), 26 (sequent calculus)

Index