CS 245 Personal Notes

by Sam Gunter

Instructors: Jonathan Buss, Collin Roberts
Textbook: Mathematical Logic for Computer Science by Zhongwan Lu
Course Notes by: Collin Roberts
• Spring 2021 • University of Waterloo •

Natural Deduction

1. Reflexivity: $A \vdash A$ is a theorem
2. $(+)$: If $\Sigma \vdash A$, then $\Sigma, \Sigma' \vdash A$ is a theorem
3. $(\neg-)$: If $\Sigma, \neg A \vdash B$ and $\Sigma, \neg A \vdash \neg B$, then $\Sigma \vdash A$ is a theorem
4. $(\to -)$: If $\Sigma \vdash A \to B$ and $\Sigma \vdash A$, then $\Sigma \vdash B$ is a theorem
5. $(\to +)$: If $\Sigma, A \vdash B$, then $\Sigma \vdash A \to B$ is a theorem
6. $(\wedge-)$: If $\Sigma \vdash A \wedge B$, then $\Sigma \vdash A$ and $\Sigma \vdash B$ are theorems
7. $(\wedge+)$: If $\Sigma \vdash A$ and $\Sigma \vdash B$, then $\Sigma \vdash A \wedge B$ is a theorem
8. $(\vee-)$: If $\Sigma, A \vdash C$ and $\Sigma, B \vdash C$, then $\Sigma, A \vee B \vdash C$ is a theorem
9. $(\vee+)$: If $\Sigma \vdash A$, then $\Sigma \vdash A \vee B$ and $\Sigma \vdash B \vee A$ are theorems
10. $(\leftrightarrow -)$: If $\Sigma \vdash A \leftrightarrow B$ and $\Sigma \vdash B$, then $\Sigma \vdash A$ is a theorem
11. $(\leftrightarrow +)$: If $\Sigma, A \vdash B$ and $\Sigma, B \vdash A$, then $\Sigma \vdash A \leftrightarrow B$ is a theorem
12. $(\forall-)$: If $\Sigma \vdash \forall x A(x)$, then for a term t, $\Sigma \vdash A(t)$ is a theorem
13. $(\forall+)$: If $\Sigma \vdash A(u)$ and $u \notin \Sigma$, then $\Sigma \vdash \forall x A(x)$ is a theorem
14. $(\exists-)$: If $\Sigma, A(u) \vdash B$ and $u \notin \Sigma \cup B$, then $\Sigma, \exists x A(x) \vdash B$ is a theorem
15. $(\exists+)$: If $\Sigma \vdash A(t)$, then for $A'(x)$ such that some occurrences of t in $A(t)$ are replaced by x, $\Sigma \vdash \exists x A'(x)$ is a theorem
16. $(\approx -)$: If $\Sigma \vdash A(t_1)$ and $\Sigma \vdash t_1 \approx t_2$, then for $A'(t_2)$ such that some occurrences of t_1 in $A(t_1)$ are replaced by t_2, $\Sigma \vdash A'(t_2)$ is a theorem
17. $(\approx +)$: $\emptyset \vDash u \approx u$ is a theorem
18. Membership Rule $(\in)$: If $A \in \Sigma$, then $\Sigma \vdash A$

Peano's Axioms

PA1: $\forall x(s(x) \not\approx 0)$

PA2: $\forall x \forall y((s(x) \approx s(y)) \to (x \approx y))$

PA3: $\forall x(x + 0 \approx x)$

PA4: $\forall x \forall y((x + s(y)) \approx s(x + y))$

PA5: $\forall x(s(x) \cdot 0 = 0)$

PA6: $\forall x \forall y(x \cdot s(y) = x \cdot y + x)$

PA7: Axiom Schema: For a formula $A(u)$ where u is a free variable,

$$(A(0) \wedge \forall x(A(x) \to A(s(x)))) \to \forall x A(x)$$

1 Introduction

Def Logic: From the Greek word Logykos, reasoning. The science of proof, the fundamental science of thought, the art of applying knowledge, the analysis of arguments

- Plato (429-327BC) studies "What is Truth"
- Aristotle (384-322BC) lays down the first systemic treatment of valid reasoning "syllogisms"
- Rene Descartes (1596-1650) introduces algebraic symbols into geometry
- George Boole (1815-1864) proposes the use of algebra for logic
- Kurt Goel (1906-1978) demonstrates that any theory must have limitations
- Alan Turing (1912-1954) gives the first definition of computer prorgamming

Def Syllogism: A kind of logical argument where a proposition is inferred from two others (premises)

Note: Based on form, not content

Applications:

1. Electronic digital circuits formed from logic gates to minimize components
2. Artificial Intelligence uses knowledge base and an inference engine
   (a) DENDRAL to identify unknown organic molecules
   (b) MYCIN to treat blood infections to the same degree as a specialist in blood infections
   (c) MISTRAL to monitor dam safety
3. Automated Proof Verifiers
4. Databases such as SQL use first order logic
5. Software Design/Programming Languages
6. DNA Computing
7. Synthetic Biology

Def Truth-Preserving: If the premises are true, then the conclusion must be true

Def Logic: The analysis and appraisal of arguments

Def Argument: A set of statements including premise(s) and a conclusion

Def Valid: An argument is valid, correct, or sound when if its premises are true its conclusion is true

Def Compound Statements: Consists of several parts, each of which is its own statement

Def Hypothetical Syllogism: If p then q, if q then r, therefore if p then r

Def Disjunctive Syllogism: p or q, not q, therefore p

Def Modus Ponens: If p then q, p, therefore q

Def Proposition: A statement with a true or false value
Variable: any variable $(\mathrm{p}, \mathrm{q}, \mathrm{r})$
Constants: 1 or true and 0 or false

Def Atomic Propositions: Cannot be further divided, a single variable

Def Compound Propositions: Combination of several variables

Def Logical Connectives: or, and, not, if, then, equivalence
Unary Connective: not or negation
Binary Connective: or, and, if, then, equivalence
Symmetric: or, and, equivalence

Def Negation: Not "reverses" the truth value of the proposition

$$\neg \mathrm{p}$$

Def Conjunction: True if both are true, false otherwise

$$\mathrm{p} \wedge \mathrm{q}$$

Def Disjunction: False if both are false, true otherwise

$$\mathrm{p} \vee \mathrm{q}$$

Def Implication: If a proposition (antecedent) implies another (consequent)

$$\mathrm{p} \rightarrow \mathrm{q}$$

Vacuously True: If $\mathrm{p} \rightarrow \mathrm{q}$ but $\neg \mathrm{p}$, then always true

Def Equivalence: Or biconditional, a double implication

$$\mathrm{p} \leftrightarrow \mathrm{q}$$

2 Propositional Language: Syntax

Def Propositional Language $\mathcal{L}^p$: The formal language of propositional logic which is expressions from a set of symbols

- Proposition Symbols: $\mathrm{p}, \mathrm{q}, \mathrm{r}, \ldots$
- Connective Symbols: $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$
- Punctuation Symbols: $(,)$

Def Expressions: Finite strings of the allowed symbols where the length is the number of occurrences of symbols, often defined by $\mathrm{U}, \mathrm{V}, etc$ (Meta-symbols, along with $=, \neq$)
Empty Expression: ε

Def Concatenation: An expression followed by another expression UV
Note: $\mathrm{U}\varepsilon = \mathrm{U}$

Def Segment: If $\mathrm{U} = \mathrm{W}_1\mathrm{V}\mathrm{W}_2$ then V is a segment of U
if $\mathrm{V} \neq \mathrm{U}$, then a Proper Segment
if $\mathrm{U} = \mathrm{VW}$, then V is an Initial Segment
if $\mathrm{U} = \mathrm{WV}$, then V is a Terminal Segment

Def Atomic Formula: An expression with length 1 consisting of a variable

$$Atom(\mathcal{L}^p)$$

Def Formulas: The set $Form(\mathcal{L}^p)$ that meet the formation rules

- Every $Atom(\mathcal{L}^p)$ is in $Form(\mathcal{L}^p)$
- For $\mathrm{A}, \mathrm{B} \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$
  - $(\neg \mathrm{A}) \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$
  - $(\mathrm{A} \wedge \mathrm{B}) \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$
  - $(\mathrm{A} \vee \mathrm{B}) \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$
  - $(\mathrm{A} \rightarrow \mathrm{B}) \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$
  - $(\mathrm{A} \leftrightarrow \mathrm{B}) \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$

Theorem Unique Readability: Every formula of $Form(\mathcal{L}^p)$ is of exactly one of the six forms in exactly one way

Lemma Every Formula in $Form(\mathcal{L}^p)$ has an equal number of left and right parentheses

Theorem Every Formula in $Form(\mathcal{L}^p)$ is an atom or one of the six forms

Def Precedence Rules: To aid in removing brackets, the connective symbols should be read in order of

- $\neg$
- $\wedge$
- $\vee$
- $\rightarrow$
- $\leftrightarrow$

Def Scopes: For ($\neg$A), A is the scope of this negation. For (A $*'$ B), A is the left scope and B is the right scope

3 Propositional Language: Semantics

Def Semantics: The meaning of logic (where syntax is the formation)

Def Truth Value: A function that maps to true or false, $\{0, 1\}$

$$t : Atom(\mathcal{L}^p) \mapsto \{0, 1\}$$

Def Truth Table: Lists all possible truth valuations
Truth valuation: A single row

Def Value: The value of A with respect to t is

$$A^t = \begin{cases} \text{if } Atom(A), & t(p) \\ \text{if } A = (\neg B)^t, & \begin{cases} 1 & \text{if } B^t = 0 \\ 0 & \text{if } B^t = 1 \end{cases} \\ \text{if } A = (B \wedge C)^t, & \begin{cases} 1 & \text{if } B^t = C^t = 1 \\ 0 & \text{else} \end{cases} \\ \text{if } A = (B \vee C)^t, & \begin{cases} 1 & \text{if } B^t \neq 0 \neq C^t \\ 0 & \text{else} \end{cases} \\ \text{if } A = (B \rightarrow C)^t, & \begin{cases} 1 & \text{if } B^t = 0 \text{ or } C^t = 1 \\ 0 & \text{else} \end{cases} \\ \text{if } A = (B \leftrightarrow C)^t, & \begin{cases} 1 & \text{if } B^t = C^t \\ 0 & \text{else} \end{cases} \end{cases}$$

Def Satisfiable: For given formulas $\Sigma \in Form(\mathcal{L}^p)$, if and only if there exists a t such that for all formulas $\Sigma^t = 1$ is satisfied ($\Sigma^t = 0$ if at least 1 formula does not satisfy)
Note: t satisfies, Σ is satisfied

Def Tautology: A formula which is true under all truth valuations

$$A^t = 1, \forall t$$

Def Contradiction: A formula which is false under all truth valuations

$$A^t = 0, \forall t$$

Def Contingent: A formula which may be true or false depending on truth valuation

Def Law of the Excluded Middle: $\mathrm{p} \lor \neg \mathrm{p}$ is a tautology

Theorem Let A be a tautology with proposition symbols $\mathrm{p_1}, \mathrm{p_2}, \dots \mathrm{p_n}$, then replacing $\mathrm{p_i}$ with arbitrary formula B is also a tautology

Def Law of the Contradiction: $\mathrm{p} \land \neg \mathrm{p}$ is a contradiction

Def Law of identity: The final of Plato's essential thoughts, $\mathrm{p} = \mathrm{p}$

Def Tautological Consequence: For $\Sigma, A \in Form(\mathcal{L}^p)$, Σ tautologically entails A if and only if

$$\Sigma \models A : \forall t, \Sigma^t = 1 \to A^t = 1$$

$\emptyset \models A$ is true for all $A \in Form(\mathcal{L}^p)$ is A is a tautology

Def Valid:

- The argument with premises $A_1, A_2, \dots A_n$ and conclusion C is valid
- $(A_1 \land A_2 \land \dots \land A_n) \to C$ is a tautology
- $(A_1 \land A_2 \land \dots \land A_n \land \neg C)$ is a contradiction
- $(A_1 \land A_2 \land \dots \land A_n \land \neg C)$ is not satisfiable
- $\{A_1, A_2, \dots A_n, \neg C\}$ is not satisfiable
- $\{A_1, A_2, \dots A_n\} \models C$

Def Tautologically Equivalent: If $A \models B$ and $B \models A$ then

$$A \models\mid B$$

Note: Equality implies equivalence but not vice versa

Def Proof by Contradiction: Assume the contrary, reach a contradiction

Def De Morgan's Law:

$$\neg(\mathrm{p} \land \mathrm{q}) \models\mid (\neg \mathrm{p} \lor \neg \mathrm{q})$$

Def Dual De Morgan's Law:

$$\neg(\mathrm{p} \lor \mathrm{q}) \models\mid (\neg \mathrm{p} \land \neg \mathrm{q})$$

Def Contrapositives:

$$(\mathrm{p} \to \mathrm{q}) \mathrel{|\!\!=\!\!|} (\neg \mathrm{q} \to \neg \mathrm{p})$$

Def Bi-Conditional:

$$\mathrm{p} \leftrightarrow \mathrm{q} \mathrel{|\!\!=\!\!|} (\mathrm{p} \to \mathrm{q}) \wedge (\mathrm{q} \to \mathrm{p})$$

Lemma Tautological Equivalences: Let $A \mathrel{|\!\!=\!\!|} A'$ and $B \mathrel{|\!\!=\!\!|} B'$, then

1. $\neg A \mathrel{|\!\!=\!\!|} \neg A'$
2. $A \wedge B \mathrel{|\!\!=\!\!|} A' \wedge B'$
3. $A \vee B \mathrel{|\!\!=\!\!|} A' \vee B'$
4. $A \to B \mathrel{|\!\!=\!\!|} A' \to B'$
5. $A \leftrightarrow B \mathrel{|\!\!=\!\!|} A' \leftrightarrow B'$

Theorem Replaceability of Tautologically Equivalent Formulas: Let $B \mathrel{|\!\!=\!\!|} B'$, and A' be A with some instances of B replaced by B', then $A' \mathrel{|\!\!=\!\!|} A$

Theorem Duality: Let A be composed of atoms, $\neg, \wedge, \vee$, then for $\Delta(A)$ made by replacing all atoms with their negation and all $\vee$ with $\wedge$ (and vice versa), $\neg A \mathrel{|\!\!=\!\!|} \Delta(A)$

Def Fuzzy Logic: Uses real values within $[0, 1]$ to denote partial truth where the restriction to $\{0, 1\}$ coincides with classical logic, now

- $\mathrm{And}(x, y) = \min\{x, y\}$
- $\mathrm{Or}(x, y) = \max\{x, y\}$
- $\mathrm{Not}(x) = 1 - x$

4 Propositional Calculus: Essential Laws, Normal Forms

Def Tautological Equivalences:

- $(A \wedge B) \wedge C \mathrel{|\!\!=\!\!|} A \wedge (B \wedge C)$
- $1 \wedge \neg A \mathrel{|\!\!=\!\!|} 0$
- $1 \wedge 0 \mathrel{|\!\!=\!\!|} 0$

Def Removing the Connectives: Connectives are definable (or reducible) in terms of $\neg, \wedge, \vee$

- $A \to B \mathrel{|\!\!=\!\!|} \neg A \vee B$
- $A \leftrightarrow B \mathrel{|\!\!=\!\!|} (A \wedge B) \vee (\neg A \wedge \neg B) \mathrel{|\!\!=\!\!|} (\neg A \vee B) \wedge (\neg B \vee A)$

Propositional Calculus Laws By dual pairs,

- Excluded Middle Law: $A \vee \neg A \models\mid 1$
- Contradiction Law: $A \wedge \neg A \models\mid 0$
- Identity Laws: $A \vee 0 \models\mid A$, $A \wedge 1 \models\mid A$
- Domination Laws: $A \vee 1 \models\mid 1$, $A \wedge 0 \models\mid 0$
- Idempotent Laws: $A \vee A \models\mid A$, $A \wedge A \models\mid A$
- Double-Negation Laws: $\neg(\neg A) \models\mid A$
- Conductivity Laws: $A \vee B \models\mid B \vee A$, $A \wedge B \models\mid B \wedge A$
- Associativity Laws: $(A \vee B) \vee C \models\mid A \vee (B \vee C)$, $(A \wedge B) \wedge C \models\mid A \wedge (B \wedge C)$
- Distributivity Laws: $A \vee (B \wedge C) \models\mid (A \vee B) \wedge (A \vee C)$, $A \wedge (B \vee C) \models\mid (A \wedge B) \vee (A \wedge C)$
- De Morgan's Laws: $\neg(A \wedge B) \models\mid \neg A \vee \neg B$, $\neg(A \vee B) \models\mid \neg A \wedge \neg B$
- Absorption Laws: $A \vee (A \wedge B) \models\mid A$, $A \wedge (A \vee B) \models\mid A$
- Another Important Law (lec 4, pg 9): $(A \wedge B) \vee (\neg A \wedge B) \models\mid B$, $(A \vee V) \wedge (\neg A \vee B) \models\mid B$

Def Literal: Complementary literals are of the form p, ¬p

Def Disjunctive Clause: A disjunction with literals as disjuncts

$$(\mathrm{p} \vee \neg \mathrm{q} \vee \mathrm{s})$$

Def Conjunctive Clause: A conjunction with literals as conjuncts

$$(\mathrm{p} \wedge \neg \mathrm{q} \wedge \mathrm{s})$$

Def Disjunctive Normal Form: A disjunction with conjunctive clauses

$$\mathrm{p} \vee (\mathrm{q} \wedge \mathrm{r})$$

Def Conjunctive Normal Form: A conjunction with disjunctive clauses

$$\mathrm{p} \wedge (\mathrm{q} \vee \mathrm{r}) \wedge (\neg \mathrm{q} \vee \mathrm{r})$$

Theorem (lec 4 pg 23) Any formula $A \in Form(\mathcal{L}^p)$ is tautologically equivalent to some formula in disjunctive normal form

Theorem (lec 4 pg 24) Any formula $A \in Form(\mathcal{L}^p)$ is tautologically equivalent to some formula in conjunctive normal form

5 Adequate Set of Connectives

Def $n-$ary Connectives: A connective $f(\mathrm{A}_1, \mathrm{A}_2, \ldots \mathrm{A}_\mathrm{n})$ is defined by its truth-table

Note: There are 4 distinct unary connectives

Note: There are 16 distinct binary connectives

p	q	$\top$	$\vee$	$\leftarrow$	$\rightarrow$	$\mid$	p	q	XOR	$\leftrightarrow$	$\neg$q	$\neg$p	$\wedge$	f	f	$\downarrow$	$\perp$
1	1	1	1	1	1	0	1	1	0	1	0	0	1	0	0	0	0
1	0	1	1	1	0	1	1	0	1	0	1	0	0	1	0	0	0
0	1	1	1	0	1	1	0	1	1	0	0	1	0	0	1	0	0
0	0	1	0	1	1	1	0	0	0	1	1	1	0	0	0	1	0

Def Adequate: A set of connectives is adequate if they can express every truth table

Note: The standard connectives are adequate $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$

Theorem (Logic 5, pg 8): The set $\{\neg, \wedge, \vee\}$ is adequate

Corollary (Logic 5, pg 10): The sets $\{\neg, \wedge\}, \{\neg, \vee\}$, $\{\neg, \rightarrow\}$ are adequate

Def NOR: The Peirce arrow connective

p	q	p $\downarrow$ q
1	1	0
1	0	0
0	1	0
0	0	1

Def NAND: The Sheffer stroke connective

p	q	p $\mid$ q
1	1	0
1	0	1
0	1	1
0	0	1

Def Ternary Connective: The if, then, else connective

p	q	r	$\tau(\mathrm{p}, \mathrm{q}, \mathrm{r})$
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	1
0	0	0	0

6 Propositional Logic: Formal Deduction

Def Formal Deduction: A relation between a set of premises and conclusion formulas

$$\{\mathrm{A}_1, \mathrm{A}_2, \dots\} = \Sigma \vdash \mathrm{A} \text{ where } \Sigma \cup \mathrm{A} = \Sigma, \mathrm{A}$$

Def Proof: A finite sequence of sequents ($\Sigma_i \vdash \mathrm{A_i}$)
Note: A Theorem is the final sequent in a valid proof

Def Natural Deduction: A set of eleven rules (or schema)

1. Reflexivity: $\mathrm{A} \vdash \mathrm{A}$ is a theorem
2. Addition of Premises: If $\Sigma \vdash \mathrm{A}$, then $\Sigma, \Sigma' \vdash \mathrm{A}$ is a theorem
3. $\neg$ Elimination: If $\Sigma, \neg\mathrm{A} \vdash \mathrm{B}$ and $\Sigma, \neg\mathrm{A} \vdash \neg\mathrm{B}$, then $\Sigma \vdash \mathrm{A}$ is a theorem
4. $\rightarrow$ Elimination: If $\Sigma \vdash \mathrm{A} \rightarrow \mathrm{B}$ and $\Sigma \vdash \mathrm{A}$, then $\Sigma \vdash \mathrm{B}$ is a theorem
5. $\rightarrow$ Introduction: If $\Sigma, \mathrm{A} \vdash \mathrm{B}$, then $\Sigma \vdash \mathrm{A} \rightarrow \mathrm{B}$ is a theorem
6. $\wedge$ Elimination: If $\Sigma \vdash \mathrm{A} \wedge \mathrm{B}$, then $\Sigma \vdash \mathrm{A}$ and $\Sigma \vdash \mathrm{B}$ are theorems
7. $\wedge$ Introduction: If $\Sigma \vdash \mathrm{A}$ and $\Sigma \vdash \mathrm{B}$, then $\Sigma \vdash \mathrm{A} \wedge \mathrm{B}$ is a theorem
8. $\vee$ Elimination: If $\Sigma, \mathrm{A} \vdash \mathrm{C}$ and $\Sigma, \mathrm{B} \vdash \mathrm{C}$, then $\Sigma, \mathrm{A} \vee \mathrm{B} \vdash \mathrm{C}$ is a theorem
9. $\vee$ Introduction: If $\Sigma \vdash \mathrm{A}$, then $\Sigma \vdash \mathrm{A} \vee \mathrm{B}$ and $\Sigma \vdash \mathrm{B} \vee \mathrm{A}$ are theorems
10. $\leftrightarrow$ Elimination: If $\Sigma \vdash \mathrm{A} \leftrightarrow \mathrm{B}$ and $\Sigma \vdash \mathrm{B}$, then $\Sigma \vdash \mathrm{A}$ is a theorem
11. $\leftrightarrow$ Introduction: If $\Sigma, \mathrm{A} \vdash \mathrm{B}$ and $\Sigma, \mathrm{B} \vdash \mathrm{A}$, then $\Sigma \vdash \mathrm{A} \leftrightarrow \mathrm{B}$ are theorems
12. Membership Rule $\in$: If $\mathrm{A} \in \Sigma$, then $\Sigma \vdash \mathrm{A}$

Def Formal Proof: Formula A is formally deducible from Σ if and only if $\Sigma \vdash \mathrm{A}$ is generate by a finite applications of the rules of formal deduction

Def Semantics: Considers what is true or false

$$A \vDash B \text{ iff } A \rightarrow B \text{ is a tautology}$$

Def Syntax: Formulas are merely a sequence of abstract symbols

$$A \vdash B \text{ iff } \emptyset \vdash A \rightarrow B$$

Lemma Double-Negation Elimination ($\neg\neg-$): For every A,

$$\neg\neg\mathrm{A} \vdash \mathrm{A}$$

Theorem Reductio Ad Absurdum ($\neg+$): If $\Sigma, \mathrm{A} \vdash \mathrm{B}$ and $\Sigma, \mathrm{A} \vdash \neg\mathrm{B}$, then

$$\Sigma \vdash \neg\mathrm{A}$$

Lemma Contrapositive: For every A, B,

$$\mathrm{A} \to \mathrm{B} \vdash \neg\mathrm{B} \to \neg\mathrm{A}$$

Theorem Transitivity of Deducibility (Tr): Let $\Sigma \subseteq Form(\mathcal{L}^p)$, $\mathrm{A}_1, \mathrm{A}_2, \ldots \mathrm{A}_\mathrm{n} \in \mathrm{Form}(\mathcal{L}^\mathrm{p})$. If $\Sigma \vdash \mathrm{A}_\mathrm{i}$ for all $i \in \{1, 2, \ldots n\}$ and $\mathrm{A}_1, \mathrm{A}_2, \ldots \mathrm{A}_\mathrm{n} \vdash \mathrm{A}$, then

$$\Sigma \vdash \mathrm{A}$$

Def Syntactically Equivalent: When $A \dashv\vdash B$ holds

Lemma Syntactic Equivalence: If $\mathrm{A} \dashv\vdash \mathrm{A}'$ and $\mathrm{B} \dashv\vdash \mathrm{B}'$, then

i) $\neg\mathrm{A} \dashv\vdash \neg\mathrm{A}'$

ii) $\mathrm{A} \land \mathrm{B} \dashv\vdash \mathrm{A}' \land \mathrm{B}'$

iii) $\mathrm{A} \lor \mathrm{B} \dashv\vdash \mathrm{A}' \lor \mathrm{B}'$

iv) $\mathrm{A} \to \mathrm{B} \dashv\vdash \mathrm{A}' \to \mathrm{B}'$

v) $\mathrm{A} \leftrightarrow \mathrm{B} \dashv\vdash \mathrm{A}' \leftrightarrow \mathrm{B}'$

Theorem Replacement of Syntactically Equivalent Formulas (Repl): Let $\mathrm{B} \dashv\vdash \mathrm{C}$, for a given A, Let A' be constructed from the replacement of some occurances of B with C, then

$$\mathrm{A} \dashv\vdash \mathrm{A}'$$

Theorem Finiteness of Premise Set: If $\Sigma \vdash \mathrm{A}$, then $\exists \Sigma_0 \subseteq \Sigma$ such that $\Sigma_0 \vdash \mathrm{A}$

Def Soundness: Every statement that one proves should be actually correct

Def Completeness: One should be able to prove, within the system, every correct statement

Theorem Soundness and Completeness: Natural Deduction is both sound and complete for propositional logic, that is

$$\forall \Sigma, \mathrm{A},\ \text{if } \Sigma \vdash \mathrm{A},\ \text{then } \Sigma \models \mathrm{A}$$

$$\forall \Sigma, \mathrm{A},\ \text{if } \Sigma \models \mathrm{A},\ \text{then } \Sigma \vdash \mathrm{A}$$

Fallacy Denying the Antecedent: $\mathrm{p} \to \mathrm{q}, \neg\mathrm{p} \vdash \neg\mathrm{q}$

Fallacy Affirming the Consequent: $\mathrm{p} \to \mathrm{q}, \mathrm{q} \vdash \mathrm{p}$

Def Inconsistent: When there is a formula A such that for the set Σ, $\Sigma \vdash \mathrm{A}$ and $\Sigma \vdash \neg\mathrm{A}$

Lemma Inconsistent Sets: Let Σ be a set of formulas, Σ is inconsistent if and only if for every B, $\Sigma \vdash \mathrm{B}$

Def Consistent: There is no formula A such that $\Sigma \vdash \mathrm{A}$ and $\Sigma \vdash \neg\mathrm{A}$

Lemma Consistent Sets: Let Σ be a set of formulas, Σ is consistent if and only if there exists a B such that $\Sigma \nvdash \mathrm{B}$

Lemma Proof and Inconsistency: Let Σ be a set of formulas, and A is a formula, then

$$\Sigma \vdash \mathrm{A} \text{ if and only if } \Sigma \cup \{\neg\mathrm{A}\} \text{ is inconsistent}$$

$$\Sigma \nvdash \mathrm{A} \text{ if and only if } \Sigma \cup \{\neg\mathrm{A}\} \text{ is consistent}$$

Lemma Proof and Inconsistency Semantic Analogue: Let Σ be a set of formulas, and A is a formula, then

$$\Sigma \vDash \mathrm{A} \text{ if and only if } \Sigma \cup \{\neg\mathrm{A}\} \text{ is unsatisfiable}$$

$$\Sigma \nvDash \mathrm{A} \text{ if and only if } \Sigma \cup \{\neg\mathrm{A}\} \text{ is satisfiable}$$

Def Maximal: A consistent set such that for every formula A, either $\mathrm{A} \in \Sigma$ or $\neg\mathrm{A} \in \Sigma$

Lemma Maximal Consistent Sets: Let Σ be a maximal consistent set, then for every A,

$$\Sigma \vdash \mathrm{A} \Longleftrightarrow \mathrm{A} \in \Sigma$$

Theorem Soundness of Propositional Formal Deduction: For all Σ, C, if $\Sigma \vdash \mathrm{C}$, then $\Sigma \vDash \mathrm{C}$

Theorem Soundness of Propositional Formal Deduction 2: If Σ is satisfiable, then Σ is consistent

Theorem Completeness of Propositional Formal Deduction: For all Σ, A, if $\Sigma \vDash \mathrm{A}$, then $\Sigma \vdash \mathrm{A}$

Lemma Maximal Consistent: Let M be a maximal consistent set $\mathrm{M} = \cup_{\mathrm{i}=0}^{\infty}\Sigma_{\mathrm{i}}$ for proposition formulas such that $\Sigma_{i+1} = \Sigma_i \cup \{\mathrm{A_i}\}$ if $\Sigma_i \cup \{\mathrm{A_i}\}$ is consistent

MC1: For every A, either $\mathrm{A} \in \mathrm{M}$ or $\neg\mathrm{A} \in \mathrm{M}$ but not both

MC2: If $\mathrm{M} \vdash \mathrm{A}$, then $\mathrm{A} \in \mathrm{M}$

MC3: For every A, B, if $\mathrm{A} \in \mathrm{M}$ and $\mathrm{A} \to \mathrm{B} \in \mathrm{M}$, then $\mathrm{B} \in \mathrm{M}$

MC4: For every $\mathrm{B} \in \mathrm{M}$, for every A, $\mathrm{A} \to \mathrm{B} \in \mathrm{M}$

MC5: For every $\mathrm{A} \notin \mathrm{M}$, for every B, $\mathrm{A} \to \mathrm{B} \in \mathrm{M}$

7 Propositional Logic: Resolution

Def Resolution: A method to to prove a disjunctive clause by proving that $\{\mathrm{A_1, A_2, \ldots A_n, \neg C}\}$ is not satisfiable (leads to a contradiction) using the formal deduction rule with parents clauses $(\mathrm{C \vee p, D \vee \neg p})$ and the resolvent $(\mathrm{C \vee D})$. Two clauses can be resolved if and only if they contain complementary literals $(\mathrm{p, \neg p})$ in which case they resolve over p

$$\mathrm{C \vee p, D \vee \neg p \vdash_r C \vee D}$$

Empty Clause: The resolvent of $\mathrm{p, \neg p \vdash_r \{\}}$

Def Resolution Derivation: From a set of clauses S, the finite set of clauses such that each clause is in S or results from S

Def Resolution Procedure: For $\mathrm{S = \{D_1, D_2, \ldots D_m\}}$, choose two clauses that contain $\mathrm{p, \neg p}$ and add their resolvent to S

Theorem Soundness of Resolution: The resolvent is tautologically implied by its parent clauses, thus it is sound

Def Set-of-Support Strategy: The set is partitioned into a non-contradictory auxiliary set and a set of support, thus each resolution takes at least one clause from the set of support

Theorem Completeness of the Set-of-Support: resolution with the set-of-support strategy is complete

Theorem Pigeonhole Principle $\mathrm{P_n}$: One cannot put $n+1$ objects into n slots with distinct objects in distinct slots

Def Davis-Putnam Procedure: Treats clauses as sets, which allows the resolvent to be the union of two clauses with $\mathrm{p, \neg p}$ omitted. A empty clause indicates a contradiction, where an empty set of clauses indicates the theorem is not valid.

- $\mathrm{S_i'}$ is all $\mathrm{S_i}$ after discarding those in which a literal and its complement appear
- $\mathrm{T_i}$ are parent clauses in which $\mathrm{p_i, \neg p_i}$ appear
- $\mathrm{U_i}$ are resolvent clauses by resolving every pair of clauses in $\mathrm{T_i}$
- $\mathrm{S_{i+1}'}$ is $\mathrm{(S_i' \setminus T_i) \cup U_i}$

Theorem Soundness and Completeness of DPP: For a finite set of clauses S, S is unsatisfiable if and only if the output of DPP is the empty clause $\{\}$

10 First-Order-Logic

Def First-Order-Logic: An extension of propositional logic to allow identification of individuals and their properties

Def Domain: The domain, or universe of discourse, is the non-empty collection of all individuals/objects that are under consideration

Def Relations: Relations, or predicates, are the properties of individuals in an argument list

Def Arity: The number of elements in the argument list of a n-ary relation
Property: A $n = 1$ relation

Def Atomic Formula: A relation name followed by an argument list that may take true/false values

$$\text{Human}(u) = \begin{cases} 1, & \text{if } u \text{ is a human} \\ 0, & \text{otherwise} \end{cases}$$

Def Quantifiers: For a quantifier (universal $\forall$x or existential $\exists$x) and scope (A(x)), the variable x is bound to the quantifier (local to the scope)

Def Free Variable: A non-bound variable

Def Quantification Over a Subset: For the universal quantifier, define an implication. For the existential quantifier, define a conjunction.

11 First-Order-Logic: Syntax

Def Language: Specifies the basic elements

Def Terms: The syntactic expressions to denote objects

Def Atomic Formulas: Combine terms into propositions

Def Formulas: Built from atomic formulas through the use of connectives and quantifiers

Def Logical Symbols: Have a fixed syntactic use and semantic meaning
Parameters: Designated syntax, but undefined semantic meaning

Def Formal Language: The formula language of first-order logic, or $\mathcal{L}$, consists of expressions using logical symbols and parameters

- Connectives: $\neg, \rightarrow, \wedge, \vee, \leftrightarrow$
- Free Variable Symbols: $\mathrm{u}, \mathrm{v}, \mathrm{w}, \mathrm{u}_1, \ldots$
- Bound Variable Symbols: $\mathrm{x}, \mathrm{y}, \mathrm{z}, \mathrm{x}_1, \ldots$

- Quantifiers: $\exists, \forall$
- Punctuation Symbols: (),
- Individual Symbols (constant symbols): $\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{a}_1, \ldots$
- Relation Symbols (predicate symbols): $\mathrm{F}, \mathrm{G}, \mathrm{H}, \mathrm{F}_1, \ldots$
- Function Symbols: $\mathrm{f}, \mathrm{g}, \mathrm{h}, \mathrm{f}_1, \ldots$

Def Equality Symbol: A special binary relation which may be contained in $\mathcal{L}$,

$$\approx$$

Def Terms: Expressions to denote objects and their properties, such that $Term(\mathcal{L})$ is closed under the following rules

1. Closed Terms: Every individual symbol is a term of $\mathcal{L}$
2. Every free-variable symbol is a term of $\mathcal{L}$
3. For terms of $\mathcal{L}$ $\mathrm{t}_1, \mathrm{t}_2, \ldots \mathrm{t}_\mathrm{n}$, the n-ary function $\mathrm{f}(\mathrm{t}_1, \mathrm{t}_2, \ldots \mathrm{t}_\mathrm{n})$ is a term of $\mathcal{L}$

Def Atoms: An expressions is in $Atom(\mathcal{L})$ if it follows one of the forms

1. If F is an $n-$ary formula symbol and $\mathrm{t}_1, \mathrm{t}_2, \ldots \mathrm{t}_\mathrm{n}$ are terms in $Term(\mathcal{L})$, then $\mathrm{F}(\mathrm{t}_1, \ldots \mathrm{t}_\mathrm{n})$ is an atom
2. If $\mathrm{t}_1, \mathrm{t}_\mathrm{n}$ are terms in $Term(\mathcal{L})$, then $\approx (\mathrm{t}_1, \mathrm{t}_2)$ is an atom

Def Formulas: An expressions in $Form(\mathcal{L})$ is closed under the following rules

1. Every atom in $Atom(\mathcal{L})$ is in $Form(\mathcal{L})$
2. If A is in $Form(\mathcal{L})$, then $(\neg \mathrm{A})$ is in $Form(\mathcal{L})$
3. If A, B are in $Form(\mathcal{L})$, then $(\mathrm{A} \vee \mathrm{B}), (\mathrm{A} \wedge \mathrm{B}), (\mathrm{A} \rightarrow \mathrm{B}), (\mathrm{A} \leftrightarrow \mathrm{B})$ are in $Form(\mathcal{L})$
4. If $\mathrm{A}(\mathrm{u})$ is in $Form(\mathcal{L})$ with u as a free-variable and x as a bound variable not in $\mathrm{A}(\mathrm{u})$, then $\forall \mathrm{x} \mathrm{A}(\mathrm{x})$ and $\exists \mathrm{x} \mathrm{A}(\mathrm{x})$ are in $Form(\mathcal{L})$

Theorem: Any term is exactly one of an individual symbol, a free variable symbol, or $f(\mathrm{t}_1, \ldots \mathrm{t}_\mathrm{n})$ where f is a n-ary function symbol

Theorem: Any formula in $Form(\mathcal{L})$, is exactly one of $(\neg \mathrm{A}), (\mathrm{A} \vee \mathrm{B}), (\mathrm{A} \wedge \mathrm{B}), (\mathrm{A} \rightarrow \mathrm{B}), (\mathrm{A} \leftrightarrow \mathrm{B}), \forall \mathrm{x} \mathrm{A}(\mathrm{x}), \exists \mathrm{x} \mathrm{A}(\mathrm{x})$

Def Closed Formula: A sentence of $Form(\mathcal{L})$ has no free-variable symbols, denoted by $Sent(\mathcal{L})$

12 First-Order-Logic: Semantics

Def Assignment: A function which assigns a value in the domain for each free variable

Def Interpretation: For the language $\mathcal{L}$, contains a non-empty domain D and a specification of each individual symbol, each relation symbol, each function symbol

Def Valuation: An interpretation and an assignment, for valuation v, the value of each term t under v is

$$\mathrm{t}^v = \begin{cases} \mathrm{c}^v & \text{if t is a constant c} \\ \mathrm{u}^v & \text{if t is a free variable u} \\ f^v(\mathrm{r}_1^v, \dots \mathrm{t}_n^v) & \text{if t is } f(\mathrm{r}_1, \dots \mathrm{t}_n) \end{cases}$$

Def Value: The value of a formula A under valuation v is

- If A is an atom, then $\mathrm{A}^v = 1$ if and only if $\langle \mathrm{t}_1^v, \dots \mathrm{t}_n^v \rangle \in \mathrm{F}^v$
- If A is a connective, then A^v is the same as in propositional logic
- If A is a quantifier, then $\forall$x must hold for every value in the domain, and $\exists$x must hold for a value within the domain

Def Re-Assigned Valuation: The valuation v with free variable u re-assigned to domain element d is

$$\mathrm{w}^{v(\mathrm{u}/d)} = \begin{cases} d & \text{if w is u} \\ \mathrm{w}^v & \text{if w is not u} \end{cases}$$

Def Quantified Formulas: The valuation v with domain D of quantified formulas is

$$(\forall \mathrm{x} \mathrm{A}(\mathrm{x}))^v = \begin{cases} 1 & \text{if } \mathrm{A}(\mathrm{u})^{v(\mathrm{u}/d)} = 1 \text{ for every } d \text{ in } D \\ 0 & \text{otherwise} \end{cases}$$

$$(\exists \mathrm{x} \mathrm{A}(\mathrm{x}))^v = \begin{cases} 1 & \text{if } \mathrm{A}(\mathrm{u})^{v(\mathrm{u}/d)} = 1 \text{ for some } d \text{ in } D \\ 0 & \text{otherwise} \end{cases}$$

Def Valid: A formula A is valid if every interpretation and valuation satisfy A ($\mathrm{A}^v = 1$), a set of formulas in $\mathcal{L}$, Σ, is valid iff for every valuation v and interpretation $\mathcal{I}$, $\Sigma^v = 1$

Def Satisfiable: A formula A is satisfiable if some interpretation and valuation satisfy A, a set of formulas in $\mathcal{L}$, Σ, is satisfiable iff there is some interpretation $\mathcal{I}$ such that $\Sigma^v = 1$

Lemma Relevance Lemma: Let A be a formula with valuations v_1, v_2 such that $\mathrm{u}^{v_1} = \mathrm{u}^{v_2}$ for every free u in A, then

$$\mathrm{A}^{v_1} = 1 \text{ if and only if } \mathrm{A}^{v_2} = 1$$

Def Logical Consequence: The set of formulas Σ entails the formula A if for every valuation v, $\Sigma^v = 1$ implies $A^v = 1$

$\emptyset \vDash A$ means A is valid

Theorem There is no algorithm for deciding the validity or satisfiability of formulas in $\mathcal{L}$

14 First-Order-Logic: Formal Deduction

Def Quasi-Formula: A formula A(u) where the free variable u is replaced by the bound variable x

Def Substitution: The process of replacing a free variable u by a term t

Def Natural Deduction for Predicate Logic: For a set of formulas $\Sigma \in \mathcal{L}$ with $A \in \mathcal{L}$, A is formally deducible from Σ if and only if the sequent $\Sigma \vdash A$ can be generated from

1. Reflexivity: $A \vdash A$ is a theorem
2. $(+)$: If $\Sigma \vdash A$, then $\Sigma, \Sigma' \vdash A$ is a theorem
3. $(\neg-)$: If $\Sigma, \neg A \vdash B$ and $\Sigma, \neg A \vdash \neg B$, then $\Sigma \vdash A$ is a theorem
4. $(\rightarrow -)$: If $\Sigma \vdash A \rightarrow B$ and $\Sigma \vdash A$, then $\Sigma \vdash B$ is a theorem
5. $(\rightarrow +)$: If $\Sigma, A \vdash B$, then $\Sigma \vdash A \rightarrow B$ is a theorem
6. $(\wedge-)$: If $\Sigma \vdash A \wedge B$, then $\Sigma \vdash A$ and $\Sigma \vdash B$ are theorems
7. $(\wedge+)$: If $\Sigma \vdash A$ and $\Sigma \vdash B$, then $\Sigma \vdash A \wedge B$ is a theorem
8. $(\vee-)$: If $\Sigma, A \vdash C$ and $\Sigma, B \vdash C$, then $\Sigma, A \vee B \vdash C$ is a theorem
9. $(\vee+)$: If $\Sigma \vdash A$, then $\Sigma \vdash A \vee B$ and $\Sigma \vdash B \vee A$ are theorems
10. $(\leftrightarrow -)$: If $\Sigma \vdash A \leftrightarrow B$ and $\Sigma \vdash B$, then $\Sigma \vdash A$ is a theorem
11. $(\leftrightarrow +)$: If $\Sigma, A \vdash B$ and $\Sigma, B \vdash A$, then $\Sigma \vdash A \leftrightarrow B$ is a theorem
12. $(\forall-)$: If $\Sigma \vdash \forall x A(x)$, then for a term t, $\Sigma \vdash A(t)$ is a theorem
13. $(\forall+)$: If $\Sigma \vdash A(u)$ and $u \notin \Sigma$, then $\Sigma \vdash \forall x A(x)$ is a theorem
14. $(\exists-)$: If $\Sigma, A(u) \vdash B$ and $u \notin \Sigma \cup B$, then $\Sigma, \exists x A(x) \vdash B$ is a theorem
15. $(\exists+)$: If $\Sigma \vdash A(t)$, then for $A'(x)$ such that some occurrences of t in A(t) are replaced by x, $\Sigma \vdash \exists x A'(x)$ is a theorem
16. $(\approx -)$: If $\Sigma \vdash A(t_1)$ and $\Sigma \vdash t_1 \approx t_2$, then for $A'(t_2)$ such that some occurrences of t_1 in $A(t_1)$ are replaced by t_2, $\Sigma \vdash A'(t_2)$ is a theorem
17. $(\approx +)$: $\emptyset \vDash u \approx u$ is a theorem
18. Membership Rule $(\in)$: If $A \in \Sigma$, then $\Sigma \vdash A$

Lemma From Logic 14 page 30: Let $\mathrm{A} \dashv\vdash \mathrm{A}', \mathrm{B} \dashv\vdash \mathrm{B}', \mathrm{C}(\mathrm{u}) \dashv\vdash \mathrm{C}'(\mathrm{u})$, then

1. $\neg\mathrm{A} \dashv\vdash \neg\mathrm{A}'$
2. $\mathrm{A} \wedge \mathrm{B} \dashv\vdash \mathrm{A}' \wedge \mathrm{B}'$
3. $\mathrm{A} \vee \mathrm{B} \dashv\vdash \mathrm{A}' \vee \mathrm{B}'$
4. $\mathrm{A} \to \mathrm{B} \dashv\vdash \mathrm{A}' \to \mathrm{B}'$
5. $\mathrm{A} \leftrightarrow \mathrm{B} \dashv\vdash \mathrm{A}' \leftrightarrow \mathrm{B}'$
6. $\forall \mathrm{x}\mathrm{C}(\mathrm{X}) \dashv\vdash \forall \mathrm{x}\mathrm{C}'(\mathrm{x})$
7. $\exists \mathrm{x}\mathrm{C}(\mathrm{X}) \dashv\vdash \exists \mathrm{x}\mathrm{C}'(\mathrm{x})$

Theorem Replacement of Equivalent Formulas: Let $\mathrm{A}, \mathrm{B}, \mathrm{C} \in Form(\mathcal{L})$ with $\mathrm{B} \dashv\vdash \mathrm{C}$ and A' be the result of substituting some occurrences of B with C, then $\mathrm{A}' \dashv\vdash \mathrm{A}$

Theorem Complementation: Let A be composed of atoms of $\mathcal{L}^p, \vee, \wedge, \neg, \forall, \exists$, if A' is the result of exchanging $\vee$ with $\wedge$, $\forall$ with $\exists$, and negating all atoms, then $\mathrm{A}' \dashv\vdash \neg\mathrm{A}$

Theorem Soundness and Completeness: Let $\Sigma \subseteq Form(\mathcal{L})$ and $\mathrm{A} \in Form(\mathcal{L})$, then $\Sigma \models \mathrm{A}$ if and only if $\Sigma \vdash \mathrm{A}$.

- since $\Sigma \vdash \mathrm{A} \to \Sigma \models \mathrm{A}$, formal natural deduction for predicate logic is sound
- since $\Sigma \models \mathrm{A} \to \Sigma \vdash \mathrm{A}$, formal natural deduction for predicate logic is complete

14 First-Order-Logic: Soundness and Completeness

Theorem: The formal deductive system of Natural Deduction is sound and complete for First-Order Logic, that is for a set of formulas Σ with formula A

$$\Sigma \vdash \mathrm{A} \text{ if and only if } \Sigma \models \mathrm{A}$$

Lemma For a terms t, $s(\mathrm{u})$, and formula $\mathrm{A}(\mathrm{u})$ with free variable u and valuation v,

$$s(\mathrm{t})^v = s(\mathrm{u})^{v(\mathrm{u}/\mathrm{t}^v)}$$

and

$$\mathrm{A}(\mathrm{t})^v = \mathrm{A}(\mathrm{u})^{v(\mathrm{u}/\mathrm{t}^v)}$$

Def Herbrand Universe: The set of terms in the language $\mathcal{L}$ ($Term(\mathcal{L})$)

$$\mathcal{H} = \{\ulcorner t \urcorner : t \text{ is a term}\}$$

Note: $\ulcorner\urcorner$ is refering to the element, not the expression

15 First-Order-Logic: Resolution and Automated Theorem Provers

Def Prenex Normal Form: A formula such that all quantifiers are in the prefix and the expression B is the matrix

$$\mathrm{Q_1x_1Q_2x_2\ldots Q_nx_nB} \text{ where Q is } \forall, \exists$$

Process

1. Eliminate all occurrences of $\to$ and $\leftrightarrow$
   - $\mathrm{A} \to \mathrm{B} \models\mid \neg\mathrm{A} \vee \mathrm{B}$
   - $\mathrm{A} \leftrightarrow \mathrm{B} \models\mid (\neg\mathrm{A} \vee \mathrm{B}) \wedge (\mathrm{A} \vee \neg\mathrm{B})$
   - $\mathrm{A} \leftrightarrow \mathrm{B} \models\mid (\mathrm{A} \wedge \mathrm{B}) \vee (\neg\mathrm{A} \wedge \neg\mathrm{B})$
2. Move all negations such that they apply to only literals
   - De Morgan's Laws $\neg(\mathrm{A} \vee \mathrm{B}) \models\mid \neg\mathrm{A} \wedge \neg\mathrm{B}, \neg(\mathrm{A} \wedge \mathrm{B}) \models\mid \neg\mathrm{A} \vee \neg\mathrm{B}$
   - Double negation $\neg\neg\mathrm{A} \models\mid \mathrm{A}$
   - $\neg\exists\mathrm{xA(x)} \models\mid \forall\mathrm{x}\neg\mathrm{A(x)}$
   - $\neg\forall\mathrm{xA(x)} \models\mid \exists\mathrm{x}\neg\mathrm{A(x)}$
3. Standardize variables with the Replaceability of Bound Variable Symbols Theorem
4. Now move quantifiers to the front
   - $\mathrm{A} \wedge \exists\mathrm{xB(x)} \models\mid \exists\mathrm{x(A} \wedge \mathrm{B(x))}$ when $x \notin A$
   - $\mathrm{A} \wedge \forall\mathrm{xB(x)} \models\mid \forall\mathrm{x(A} \wedge \mathrm{B(x))}$ when $x \notin A$
   - $\mathrm{A} \vee \exists\mathrm{xB(x)} \models\mid \exists\mathrm{x(A} \vee \mathrm{B(x))}$ when $x \notin A$
   - $\mathrm{A} \vee \forall\mathrm{xB(x)} \models\mid \forall\mathrm{x(A} \vee \mathrm{B(x))}$ when $x \notin A$

Theorem Replaceability of Bound Variable Symbols: For a formula $\mathrm{A} \in Form(\mathcal{L})$, if A' results from replacing some occurrences of $\mathrm{QxB(x)}$ with $\mathrm{QyB(y)}$ then $\mathrm{A} \models\mid \mathrm{A}'$ and $\mathrm{A} \vdash\mid \mathrm{A}'$

Def $\exists$-free Prenex Normal Form: A sentence $\mathrm{A} \in Sent(\mathcal{L})$ that is in prenex normal form and contains no existential quantifier symbols.

Def Skolem Function: The function $\mathrm{f(x_1, x_2, \ldots x_n)}$ that expresses the individual generated by $\exists\mathrm{yA}$ in $\forall\mathrm{x_1}, \forall\mathrm{x_2}, \ldots \forall\mathrm{x_n}\exists\mathrm{yA}$ for given $\mathrm{x_1, x_2, \ldots x_n}$. For a $\mathrm{A} \in Sent(\mathcal{L})$ with $\mathrm{A} = \forall\mathrm{x_1}\forall\mathrm{x_2}\ldots\forall\mathrm{x_n}\exists\mathrm{yA}$, define A' to be A with all instances of y replaced by the Skolem function $\mathrm{f(x_1, x_2, \ldots x_n)}$, then the sentence A skolemized is

$$\forall\mathrm{x_1}\forall\mathrm{x_2}\ldots\forall\mathrm{x_nA}' \text{ (often wrote as } \mathrm{A}' \text{ without quantifiers)}$$

Theorem Given a $\mathrm{A} \in Sent(\mathcal{L})$, there is an efficient procedure for finding a $\exists$-free prenex normal form A' such that A is satisfiable if and only if A' is satisfiable

Theorem Given a $\mathrm{A} \in \exists$-free prenex normal form, there is an efficient procedure for finding a finite set $\mathrm{C_A}$ of disjunctive clauses such that A is satisfiable if and only if $\mathrm{C_A}$ is satisfiable

Theorem For a set $\Sigma \subseteq Sent(\mathcal{L})$ with $\mathrm{A} \in \mathrm{Sent}(\mathcal{L})$, then $\Sigma \models \mathrm{A}$ is valid if and only if the set

$$\mathrm{C}_{\neg\mathrm{A}} \cup \left[\bigcup_{\mathrm{B}\in\Sigma} \mathrm{C_B}\right]$$

is not satisfiable

Def Instantiation: An assignment of a quasi-term t'_i to a variable $\mathrm{x_i}$

$$\mathrm{x_i} := \mathrm{t}'_\mathrm{i}$$

Def Unification: Two formulas unify if there is a unifer instantiation such that the formulas are identical

Process Automated Theorem Proving:

1. Convert $\mathrm{A} \in Sent(\mathcal{L})$ to prenex normal form
2. Remove existential quantifiers and replace with Skolem functions
3. Drop the universal quantifiers
4. Convert into conjunctive normal form clauses
5. Attempt to find $\{\}$ (not satisfiable, thus valid) by resolving with unification

Theorem A set S of clauses is not satisfiable if and only if there is a resolution of the empty clause $\{\}$

- Soundness: If the resolution of S outputs the empty clause, then S is not satisfiable
- Completeness: If S is not satisfiable, then the resolution of S outputs the empty clause

Def Automated Proof Verification: Checks that a formal proof is correct, CASC is a yearly competition for first-order provers to compete

- Isabelle: higher order theorem prover using resolution and unification
- Coq: Proof checker including proving tactics and various decision procedures
- E or SPASS: Theorem prover for first order logic with equality
- Vampire: Theorem prover for first order logic

16 Computing with Logical Formulas

Def Algorithm: A finite sequence of well-defined computer-implementable instructions that solve a problem (give the correct output for every input)

- Analytical Engine: Babbage's met almost none of the requirements, it failed
- λ-Calculus: Church's had mathematical qualifications, but was not (clearly) general or implementable
- Turing Machine: An analogy using a tape (stack) of cells (paper) that could be edited (pencil) moved between (flip page) and operated on (read)

Def Turing Machine (TM): A $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ such that

- Q is the finite set of states of the control
- Σ is the finite set (alphabet) of input symbols
- Γ is the finite set of tape symbols such that $\Sigma \in \Gamma$
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the transition function with left and right

  $\delta(q, a) = (r, b, d)$ where a at state q becomes b, then steps R/L (d) to set state r

- $q_0 \in Q$ is the start (initial) state
- B is the blank symbol such that $B \in \Gamma, B \notin \Sigma$
- $F \in Q$ is the set of final (accepting) states

Def TM Outcomes: For a TM M starting at string x

- If M reaches state q and symbol a such that $\delta(q, a)$ is undefined

$$M \text{ halts on } x, \begin{cases} \text{If } q \in F, \text{ then } M \text{ accepts } x \\ \text{If } q \notin F, \text{ then } M \text{ rejects } x \end{cases}$$

- If M attempts to move left from the leftmost cell,

$$M \text{ crashes on } x$$

- If M continues making transitions forever,

$$M \text{ runs forever (loops) on } x$$

Def Decide: A TM M with alphabet Σ such that for all $x \in \Sigma$, M with x halts. Then M decides the language of strings over Σ that lead M to accept, that is

$$\{x \in \Sigma^* \mid M \text{ accepts input } x\}$$

Def Decidability: If there exists an algorithm (TM) that solves (decides) it

Def Turing Capabilities:

- Store finite information: In control or in the cells
- Make space on the tape: Using shift to free room
- Control structures: We can impose loops, recursion, subroutines, etc
- Copy a string
- Compare strings
- Lookup: An association list implements a dictionary

Def Universal Turing Machine: A TM, that given a description of another TM M, and an input x, performs M on x

Def Code: A code of a TM M, denoted $\langle M\rangle$ is the concatenate of all (i, j, i', j', d) tuples such that $\delta(q_i, s_j) = (i', j', d)$

Def Halting Problem: Given a $\langle M\rangle$ and x, does M with input x halt

$$\text{HALT } = \{\langle M\rangle, x \mid M \text{ halts on } x\}$$

Theorem Turing: The Halting Problem is undecidable

Def Reduction: A reduction from P_1 to P_2 is an always halting algorithm that given an instance x_1 of P_1, produces a x_2 of P_2 such that in every case the answer to x_1 is equal to the answer to x_2. For a reduction R,

$$\forall x_1, x_1 \in P_1 \text{ if and only if } R(x_1) \in P_2$$

Thus if P_2 is decidable, then P_1 is decidable and if P_1 is undecidable, then P_2 is undecidable

Def Post Correspondence Problem (PCP): Given a finite sequence $(s_1, t_1), (s_2, t_2) \dots (s_k, t_k)$ such that all strings are positive length, is there a sequence of indices $i_1, i_2, \dots i_n$ with $n \geq 1$ such that the concatenation of $s_{i_1} s_{i_2} \dots s_{i_n} = t_{i_1} t_{i_2} \dots t_{i_n}$ [undecidable]

Def Hilbert's Tenth Problem (IntegerRoot): Given a polynomial $q(x_1, x_2, \dots x_n)$ with integers coefficients, does q have an integral root $(q(a_1, a_2, \dots a_n) = 0)$ [undecidable]

Lemma There is a Turing Machine that given A of $Form(\mathcal{L})$, outputs a proof of A if such a proof exists, otherwise it may run forever

Theorem Godel's Incompleteness Theorem: For a set of formulas Γ such that membership is decidable, then $PA \cup \Gamma$ is inconsistent or there is a formula A such that $PA \cup \Gamma \nvdash A$ and $PA \cup \Gamma \nvdash \neg A$

17 Peano Arithmetic

Def Equality: Formal deduction rules concerning equality lead to reflexively, symmetry, and transitivity

- $(\approx +)$: $\emptyset \vdash \mathrm{u} \approx \mathrm{u}$
- $(\approx -)$: If $\Sigma \vdash \mathrm{A}(\mathrm{t}_1)$ and $\mathrm{t}_1 \approx \mathrm{t}_2$, then $\Sigma \vdash \mathrm{A}(\mathrm{t}_2)$ where $\mathrm{A}(\mathrm{t}_2)$ is $\mathrm{A}(\mathrm{t}_1)$ with some instances of t_1 replaced with t_2

Def Reflexivity: $\forall \mathrm{x}(\mathrm{x} \approx \mathrm{x})$

Def Symmetry: $\forall \mathrm{x} \forall \mathrm{y}((\mathrm{x} \approx \mathrm{y})) \to (\mathrm{y} \approx \mathrm{x}))$

Def Transitivity: $\forall \mathrm{x} \forall \mathrm{y} \forall \mathrm{z}(((\mathrm{x} \approx \mathrm{y}) \wedge (\mathrm{y} \approx \mathrm{z})) \to (\mathrm{x} \approx \mathrm{z}))$

Theorem EQSubs: For variable u and terms $\mathrm{r}, \mathrm{t}_1, \mathrm{t}_2$ where $\mathrm{r}(\mathrm{t}_i)$ is r with all instances of u replaces with t_i. If $\Sigma \vdash \mathrm{t}_1 \approx \mathrm{t}_2$, then

$$\Sigma \vdash \mathrm{r}(\mathrm{t}_1) \approx \mathrm{r}(\mathrm{t}_2)$$

Theorem EQtrans(k): For all Σ with terms $\mathrm{t}_1, \mathrm{t}_2, \ldots \mathrm{t}_{k+1}$, if $\Sigma \vdash \mathrm{t}_i \approx \mathrm{t}_{i+1}$ for each $1 \leq \mathrm{t}_i \leq k$, then

$$\Sigma \vdash \mathrm{t}_1 \approx \mathrm{t}_{k+1}$$

Def Domain Axioms: The set A of formulas that are accepted/assumed to be true within the class of domains

- Decidable: The set A should have an algorithm to determine which formulas are domain axioms
- Sound with Respect to Domain: The set A should guarantee that if $\mathrm{A} \vdash \mathrm{B}$, then B holds in the domain
- Complete: The set A should allow the formal proof of all true formulas in the domain

Def Axioms of Geometry: Euclid's Postulates appeared in Elements

1. A straight line may be drawn between any two points
2. Every straight line can be extended infinitely
3. A circle may be drawn with any center point, and any radius
4. All right angles are equal
5. Parallel Postulate: For every given point not on a given line, there is exactly one line through the point that does not meet the given line

Def Theory: The set of all formulas that can be proven by a set of axioms. For a given theory T with axioms X_T, $\Sigma \vdash_T C$ denotes $\Sigma, X_T \vdash C$

Def Number Theory Arithmetic: With the domain of $\mathbb{N}$, addition, multiplication, and ordering, we attempt to have a base set of axioms that derive all theorems

Def Peano's Axioms: Constant is 0, functions are $+, \cdot, s$, equality is $\approx$, then PA is

PA1: $\forall \mathrm{x}(\mathrm{s}(\mathrm{x}) \not\approx 0)$

PA2: $\forall \mathrm{x} \forall \mathrm{y}((\mathrm{s}(\mathrm{x}) \approx \mathrm{s}(\mathrm{y})) \rightarrow (\mathrm{x} \approx \mathrm{y}))$

PA3: $\forall \mathrm{x}(\mathrm{x} + 0 \approx \mathrm{x})$

PA4: $\forall \mathrm{x} \forall \mathrm{y}((\mathrm{x} + \mathrm{s}(\mathrm{y})) \approx \mathrm{s}(\mathrm{x} + \mathrm{y}))$

PA5: $\forall \mathrm{x}(\mathrm{s}(\mathrm{x}) \cdot 0 = 0)$

PA6: $\forall \mathrm{x} \forall \mathrm{y}(\mathrm{x} \cdot \mathrm{s}(\mathrm{y}) = \mathrm{x} \cdot \mathrm{y} + \mathrm{x})$

PA7: Axiom Schema: For a formula A(u) where u is a free variable,

$$(\mathrm{A}(0) \wedge \forall \mathrm{x}(\mathrm{A}(\mathrm{x}) \rightarrow \mathrm{A}(\mathrm{s}(\mathrm{x})))) \rightarrow \forall \mathrm{x} \mathrm{A}(\mathrm{x})$$

Def $\leq$: $\mathrm{u} \leq \mathrm{v}$ if and only if $\exists \mathrm{z}(\mathrm{u} + \mathrm{z} = \mathrm{v})$

Def $<$: $\mathrm{u} < \mathrm{v}$ if and only if $\exists \mathrm{z}(\mathrm{u} + \mathrm{s}(\mathrm{z}) = \mathrm{v})$

Def Even: $Even(\mathrm{u})$ if and only if $\exists \mathrm{y}(\mathrm{u} = \mathrm{y} + \mathrm{y})$

Def Prime: $Prime(\mathrm{u})$ if and only if $(1 < \mathrm{u}) \wedge \neg \exists \mathrm{y} \exists \mathrm{z}((\mathrm{u} = \mathrm{y} \cdot \mathrm{z}) \wedge (1 < \mathrm{y}) \wedge (1 < \mathrm{z}))$

Def Godel's Incompleteness Theorem: In any consistent formal theory T with a decidable set of axioms that expresses elementary arithmetic, there exists a statement that is true but cannot be proven

Def Paris-Harrington Theorem: One such theorem. Certain large sets exists but they are so large that Peano's Axioms cannot prove that they exist

18 Program Verification

Def Program Correctness: By inspection, testing and formal verification, does a program satisfy its specifications

Def Formal Program Verification:

1. Convert specifications R into a formula $\mathrm{A_R}$ of symbolic logic
2. Write program P which should realize $\mathrm{A_R}$ in a given programming environment
3. Prove that P satisfies $\mathrm{A_R}$

Def Imperative Programming: Defines control flow as statements that change a programs state variables (C, C++, Java, Python, FORTRAN, Pascal)

Def Declarative Programming: Focuses on what should be accomplished with less how it should be (query such as SQL, logic such as PROLOG, functional such as scheme)

Def Sequential: No concurrency

Def Transformational: given inputs, compute outputs and terminate

Def Core Programming Language:

- Integer and Boolean expressions
- Assignment
- Sequence
- if-then-else conditional statements
- while-loops
- for-loops (omitted)
- arrays
- functions and procedures (omitted)

Def Hoare Triple: Precondition, code, postcondition $(\!|\mathrm{P}|\!)\mathrm{C}(\!|\mathrm{Q}|\!)$ where the precondition can be set to true if no constraints

Def Partial Correctness: For every state s such that P is true, if the execution of C terminates in s' and s' satisfies Q if and only if

$$\models_{par} (\!|P|\!)C(\!|Q|\!)$$

Def Complete Correctness: For every state s such that P is true, the execution of C terminates to s' and s' satisfies Q if and only if

$$\models_{tot} (\!|P|\!)C(\!|Q|\!)$$

Def First-Order Logic Conditions:

- Relation State(s): s is a program state
- Relation Condit(P): P is a condition
- Relation Code(C): C is a program
- Relation Inv(I): I is an invariant
- Relation Satisfies(s, P): State s satisfies condition P
- Relation Terminates(C, s): Program C terminates when execution begins in state s
- Function result(C, s): the state that results from executing code C on state s (if it terminates)

- Partial correctness of Hoare triple $(\!| \mathrm{P} |\!)\ \mathrm{C}\ (\!| \mathrm{Q} |\!)$:

  $\forall s \forall P \forall C \forall Q [\mathrm{State}(s) \wedge \mathrm{Condit}(P) \wedge \mathrm{Code}(C) \wedge \mathrm{Condit}(Q) \longrightarrow$
  $(\mathrm{Satisfies}(s, P) \wedge \mathrm{Terminates}(C, s) \longrightarrow \mathrm{Satisfies}(\mathrm{result}(C, s), Q))]$

- Total correctness of Hoare triple $(\!| \mathrm{P} |\!)\ \mathrm{C}\ (\!| \mathrm{Q} |\!)$:

  $\forall s \forall P \forall C \forall Q [\mathrm{State}(s) \wedge \mathrm{Condit}(P) \wedge \mathrm{Code}(C) \wedge \mathrm{Condit}(Q) \longrightarrow$
  $(\mathrm{Satisfies}(s, P) \longrightarrow \mathrm{Terminates}(C, s) \wedge \mathrm{Satisfies}(\mathrm{result}(C, s), Q))]$

Def Partial Correctness Proof: Annotated program where each statement has a Hoare triple with justification

Def Logical: After implied rules have been used, ordinary logic proofs must be sued to justify

Def Total Correctness Proof: For each while-loop, identify a strictly-decreasing non-negative integer through the loop (variant)

Def Array Assignment: $\mathrm{A}\{\mathrm{i} \leftarrow \mathrm{e}\}$ is the array with entries given by

$$\mathrm{A}\{\mathrm{i} \leftarrow \mathrm{e}\}[\mathrm{j}] = \begin{cases} \mathrm{e}, & \text{if } \mathrm{j} = \mathrm{i} \\ \mathrm{A}[\mathrm{j}], & \text{if } \mathrm{j} \neq \mathrm{i} \end{cases}$$

Theorem The Total Correctness Problem is undecidable

Theorem The Partial Correctness Problem is undecidable

Def Inference Rules:

- Assignment Rule: $Q[E/x]$ is read as Q with E in place of x,

$$\frac{}{\llparenthesis \mathrm{Q[E/x]} \rrparenthesis \mathrm{x = E} \llparenthesis \mathrm{Q} \rrparenthesis}$$

- Precondition Strengthening: $\emptyset \vdash P \to P'$,

$$\frac{\mathrm{P \to P'}, \llparenthesis \mathrm{P'} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{Q} \rrparenthesis}{\llparenthesis \mathrm{P} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{Q} \rrparenthesis}$$

- Postcondition Weakening: $\emptyset \vdash Q' \to Q$,

$$\frac{\llparenthesis \mathrm{P} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{Q'} \rrparenthesis, \mathrm{Q' \to Q}}{\llparenthesis \mathrm{P} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{Q} \rrparenthesis}$$

- Composition: Implicit,

$$\frac{\llparenthesis \mathrm{P} \rrparenthesis \mathrm{C_1} \llparenthesis \mathrm{Q} \rrparenthesis, \llparenthesis \mathrm{Q} \rrparenthesis \mathrm{C_2} \llparenthesis \mathrm{R} \rrparenthesis}{\llparenthesis \mathrm{P} \rrparenthesis \mathrm{C_1, C_2} \llparenthesis \mathrm{R} \rrparenthesis}$$

- If-Then-Else:

$$\frac{\llparenthesis \mathrm{P \land B} \rrparenthesis \mathrm{C_1} \llparenthesis \mathrm{Q} \rrparenthesis, \llparenthesis \mathrm{P \land \neg B} \rrparenthesis \mathrm{C_2} \llparenthesis \mathrm{Q} \rrparenthesis}{\llparenthesis \mathrm{P} \rrparenthesis \text{ if } \mathrm{(B), C_1} \text{ else } \mathrm{C_2} \llparenthesis \mathrm{Q} \rrparenthesis}$$

- If-Then:

$$\frac{\llparenthesis \mathrm{P \land B} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{Q} \rrparenthesis, \mathrm{(P \land \neg B) \to Q}}{\llparenthesis \mathrm{P} \rrparenthesis \text{ if } \mathrm{(B), C} \llparenthesis \mathrm{Q} \rrparenthesis}$$

- Partial-While: With loop invariant I,

$$\frac{\llparenthesis \mathrm{I \land B} \rrparenthesis \mathrm{C} \llparenthesis \mathrm{I} \rrparenthesis}{\llparenthesis \mathrm{I} \rrparenthesis \text{ while } \mathrm{(B), C} \llparenthesis \mathrm{I \land \neg B} \rrparenthesis}$$

- Array-Assignment: With $\mathrm{A\{i \leftarrow e\}[j]} = \begin{cases} \mathrm{e}, & \text{if } \mathrm{j = i} \\ \mathrm{A[j]}, & \text{if } \mathrm{j \neq i} \end{cases}$,

$$\frac{}{\llparenthesis \mathrm{Q[A\{e_1 \leftarrow e_2\}/A]} \rrparenthesis \mathrm{A[e_1] = e_2} \llparenthesis \mathrm{Q} \rrparenthesis}$$