

Week-4
lecture16

This file, "lecture16.pdf," focuses on **Propositional Logic (PL)**. It covers the fundamental concepts, syntax, semantics, and rules of inference within this logical framework.

Here's a breakdown of the key topics:

- **Propositional Logic Basics:**
 - **Truth Values:** Introduces the core idea that statements (propositions) can be either true or false.
 - **Sentential Connectives:** Explains how simple propositions are combined using "and" (\wedge), "or" (\vee), "not" (\neg), "implies" (\rightarrow), and "equivalent" (\leftrightarrow).
 - **Truth Conditions:** Defines when compound propositions formed with these connectives are true or false (e.g., $X \wedge Y$ is true if both X and Y are true).
- **Syntax and Structure:**
 - **Symbols:** Outlines the basic symbols of PL, including truth, false, propositional symbols (P, Q), logical connectives, and parentheses.
 - **Sentence Formation Rules:** Describes how to construct well-formed sentences using these symbols and connectives.
 - **Connectives:** Provides detailed definitions and examples for negation, conjunction, disjunction, implication, and biconditional.
 - **Backus-Naur Form (BNF):** Introduces a formal grammar for propositional logic, demonstrating how sentences are structured.
 - **Operator Precedence:** Establishes the order in which logical operators are evaluated ($\neg, \wedge, \vee, \rightarrow, \leftrightarrow$).
- **Semantics and Truth Tables:**
 - **Interpretation:** Explains how meaning is given to symbols, with sentences being either true or false.
 - **Truth Tables:** Illustrates the truth values of complex sentences based on the truth values of their atomic parts. A truth table for the five main logical connectives is provided.
 - **Validity/Tautology:** Defines a valid sentence (tautology) as one that is true in every possible interpretation, with an example.
- **Rules of Inference and Logical Equivalences:**
 - **Theorem Proving:** Presents rules of inference used for proving theorems:
 - **Modus Ponens:** If α implies β , and α is true, then β can be inferred.
 - **And-Elimination:** From a conjunction ($\alpha \wedge \beta$), either α or β can be inferred.
 - **Inference with Biconditional:** Rules for converting between $\alpha \leftrightarrow \beta$ and $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$.
 - **Resolution:** A rule for inferring a disjunction from two clauses containing complementary literals.
 - **Monotonicity:** States that adding new sentences to a knowledge base does not invalidate previous conclusions.
 - **Laws of PL/Logical Equivalences:** Lists and defines common logical equivalences such as double negation, contrapositive, De Morgan's laws, commutative, associative, and distributive laws.
- **Resolution Refutation and CNF:**
 - **Procedure:** Describes the steps for the Resolution Refutation procedure to prove a well-formed formula (WFF). This involves converting WFFs to Conjunctive Normal

Form (CNF) and applying resolution until an empty clause is produced (indicating the WFF is entailed).

- **Conversion to CNF:** Outlines the steps to convert a sentence into Conjunctive Normal Form (CNF) by eliminating implications, reducing the scope of negation, and applying associative and distributive laws.
- **Example Problem:** Provides a step-by-step example of converting WFFs to CNF and demonstrating how to show if a conclusion can be entailed from a knowledge base.
- **Problems with PL:**
 - The lecture concludes by highlighting limitations of propositional logic, such as the need for numerous propositions to represent complex scenarios (e.g., time-dependent events or statements about multiple instances of the same variable). It briefly introduces **First Order Logic (FOL)** as a solution to these problems, mentioning its use of objects, relations, and quantifiers.

Lecture17

This lecture, "AD Predicate Logic - 1" by V. Susheela Devi, introduces First-Order Logic (FOL) as an advancement over Propositional Logic, addressing its limitations in representing complex environments with many objects.

Here's a summary of the key points:

- **Limitations of Propositional Logic:** It has a limited ontology, making it difficult to represent complex information concisely, especially in environments with many objects.
- **Introduction to First-Order Logic (FOL):** FOL makes stronger ontological commitments, consisting of objects, properties, relations, and functions, allowing for a more expressive language similar to natural languages.
- **Components of FOL:**
 - **Objects:** Nouns and noun phrases (e.g., people, houses, Ronald McDonald).
 - **Relations:** Verbs and verb phrases describing connections between objects. These can be unary (properties like "red") or n-ary (like "brother of," "bigger than").
 - **Functions:** Special types of relations that map arguments to a single object (e.g., "father of," "one more than").
- **Syntax of FOL:**
 - **Symbols:** Constant symbols (for objects), Predicate symbols (for relations), and Function symbols (for functions).
 - **Terms:** Logical expressions referring to an object, which can be simple (constants, variables) or complex (formed by function symbols and terms).
 - **Atomic Sentences:** State facts, formed by a predicate symbol followed by a parenthesized list of terms (e.g., `Brother(Rama, Lakshman)`).
 - **Complex Sentences:** Constructed using logical connectives (`AND`, `OR`, `IMPLIES`, `NOT`).
 - **Quantifiers:** Used to express properties of collections of objects.
 - **Universal Quantification (\forall):** "For all" (e.g., `$\forall x$ Crow(x) \rightarrow Black(x)`).
 - **Existential Quantification (\exists):** "There exists" (e.g., `$\exists x$ Sister(x, Spot) \wedge Cat(x)`).
 - **Nested Quantifiers:** Combining multiple quantifiers for complex

statements.

- **Connection Between \forall and \exists** : They are intimately linked through negation (e.g., $\forall x \neg P \equiv \neg \exists x P$). Both are used for readability.
- **Equality Symbol ($=$)**: Used to signify that two terms refer to the same object. It can also be used with negation (\neq) to assert that two terms are not the same.
- **Higher Order Logic (HOL)**: Briefly introduced as an extension of FOL where quantification is allowed over relations and functions themselves, providing more expressive power.
- **Using First-Order Logic**: Sentences are added to a knowledge base using **TELL** (assertions) and queries are made using **ASK**. **ASKVARS** is used to find specific substitutions or binding lists for variables in quantified queries.

The lecture also includes a table classifying formal languages by their ontological and epistemological commitments, and a Backus-Naur form diagram illustrating the syntax of first-order logic with equality.

Lecture18

This file, "lecture18.pdf," is lecture notes that cover advanced topics in predicate logic, focusing on how to represent complex concepts and relationships using logical expressions.

The key areas covered are:

- **Family Relationships/Kinships**: This section details how to represent family structures using binary predicates (e.g., Parent, Sibling, Spouse) and unary operators (Male, Female). It also provides axioms for defining relationships like Mother, Husband, Grandparent, and Sibling.
- **Numbers**: This part introduces how natural numbers can be defined recursively using the predicate **NatNum**, a constant **0**, and a successor function **S**. It also shows how addition can be defined in terms of the successor function, with mentions of multiplication, integer division, and prime numbers.
- **Sets**: This section covers the logical representation of set theory concepts. It introduces the empty set **{}**, a unary predicate **Set**, binary predicates for membership ($x \in s$) and subset ($s1 \subseteq s2$), and binary functions for intersection ($s1 \cap s2$), union ($s1 \cup s2$), and adjoining an element ($\{x | s\}$). Several axioms are provided to define how sets are formed, their elements, subsets, equality, intersection, and union.
- **Tower of Hanoi**: This section briefly applies predicate logic to the classic Tower of Hanoi puzzle. It defines predicates like **Smaller**, **Free**, and **Legal** to represent the rules and conditions for moving disks.
- **Uniqueness Operator**: The file introduces the uniqueness operator ($\exists!$), used to denote that there is exactly one element satisfying a given condition.
- **Example Sentences**: A significant portion of the document is dedicated to providing numerous examples of translating English sentences into predicate logic expressions. These examples cover various scenarios, including existential and universal quantification, negation, and complex logical structures, ranging from simple statements about students and courses to more abstract ideas about loyalty, policies, and personal attributes.

Lecture19

This document, "Lecture 19: Inference in Predicate Logic," by V. Susheela Devi, covers

advanced inference techniques in First-Order Logic (FOL).

The main topics include:

- **Extension of Propositional Logic Inference Rules:** It explains how inference rules from propositional logic (Modus Ponens, And-Elimination, And-Introduction, Or-Introduction, Resolution) also apply to FOL, with additional rules for quantifiers: Universal Elimination, Existential Elimination, and Existential Introduction.
- **Proposition vs. FOL:** The lecture discusses how FOL can be converted to propositional logic for inference. It details Universal Instantiation (substituting ground terms for variables) and Existential Instantiation (introducing Skolem constants to name existentially quantified objects).
- **Generalized Modus Ponens:** This section introduces a generalized version of Modus Ponens that works with first-order clauses by finding appropriate substitutions to match premises.
- **Unification:** A core concept, unification is explained as the process of finding substitutions that make different logical expressions identical. The UNIFY algorithm is introduced, along with examples and a discussion of standardizing apart variables to avoid clashes. The document includes a detailed algorithm for computing most general unifiers.
- **Forward Chaining:** This inference method starts with known facts and applies Modus Ponens to derive new facts until no more inferences can be made or a query is answered. The lecture provides an example of proving "West is a criminal" by converting the problem into FOL clauses and demonstrating a simple forward-chaining algorithm (FOL-FC-ASK), including its soundness and completeness for definite clause knowledge bases.
- **Efficient Forward Chaining:** The document also addresses inefficiencies in forward chaining, such as the "inner loop" for pattern matching, rechecking rules, and generating irrelevant conclusions. Solutions discussed include the conjunct ordering problem (e.g., using the minimum-remaining-value heuristic), incremental forward chaining, and using "magic sets" to restrict inferences to relevant variable bindings.

In essence, this lecture provides a comprehensive overview of how inference is performed in First-Order Logic, moving from foundational rules to practical algorithms like unification and forward chaining, and addressing their computational challenges.

Lecture20

This file, "lecture20.pdf," focuses on **Inference in First-Order Logic (FOL)**, specifically covering forward chaining, backward chaining, and resolution.

Here's a breakdown of the key topics:

- **Forward Chaining (Pages 2-6):**
 - Introduces a simple forward-chaining algorithm (FOL-FC-ASK) which starts with known facts and repeatedly applies rules to derive new conclusions until a query is answered or no new facts can be added.
 - Discusses the soundness and completeness of FOL-FC-ASK for definite clause knowledge bases.
 - Highlights inefficiencies in forward chaining, such as the "matching rules against

known facts" problem, rechecking every rule, and generating irrelevant conclusions.

- Presents strategies to improve efficiency, including the conjunct ordering problem (e.g., minimum-remaining-value heuristic), incremental forward chaining, and avoiding irrelevant conclusions (e.g., backward chaining, restricting rules, magic sets).
- **Backward Chaining (Pages 7-9):**
 - Explains that backward-chaining algorithms work by starting from the goal and chaining backward through rules to find supporting known facts.
 - Provides a simple backward-chaining algorithm (FOL-BC-ASK) which is implemented as a generator to return multiple possible results.
 - Illustrates a proof tree generated by backward chaining to prove a "criminal" example.
- **Resolution (Pages 10-18):**
 - Introduces resolution as an inference rule for first-order logic, requiring sentences to be in Conjunctive Normal Form (CNF).
 - Outlines the steps to convert a sentence into CNF, including eliminating implications, moving negation inwards, standardizing variables, Skolemization (removing existential quantifiers), dropping universal quantifiers, and distributing disjunction over conjunction.
 - Demonstrates the resolution inference rule with examples.
 - Provides two detailed example proofs using resolution:
 - Proving "Criminal(West)" from a given knowledge base (crime example).
 - Proving "Curiosity killed the cat" (animal and killing example), with a paraphrase of the logical steps.
- **Resolution Strategies (Pages 19-21):**
 - Discusses strategies to find resolution proofs more efficiently:
 - **Unit Preference:** Prioritizes resolutions involving unit clauses (single literals) to produce shorter clauses.
 - **Set of Support:** Insists that every resolution step involve at least one element from a special "set of support" (often the negated query) to reduce the search space and generate goal-directed proof trees.
 - **Input Resolution:** Combines an input sentence (from KB or query) with another sentence, complete for Horn clauses but not general cases.
 - **Linear Resolution:** A generalization of input resolution that allows combining sentences if one is from the KB or an ancestor in the proof tree (complete).
 - **Subsumption:** Eliminates sentences that are more specific than existing sentences in the KB to keep the KB small.
- **Homework Example (Page 22):**
 - Presents a problem in English (about Marcus, Pompeii, mortality, and a volcano) and asks to prove that "Marcus is not alive now," implying that the concepts discussed in the lecture are to be applied to solve this problem.

In essence, "lecture20.pdf" serves as a comprehensive guide to three fundamental inference mechanisms in First-Order Logic, detailing their algorithms, efficiency considerations, and application in constructing proofs.