Knowledge-Based Systems and Deductive Databases

Wolf-Tilo Balke
Jan-Christoph Kalo
Institut für Informationssysteme
Technische Universität Braunschweig
http://www.ifis.cs.tu-bs.de
2. Logics for Knowledge Bases

2.0 Introduction to Logics
2.1 Syntax of First Order Logic
2.2 Semantics of First Order Logic
2.0 Goals Of this lecture

• Learn something about the **history** of logics
  – Ideas and people behind logic
  – Come (hopefully) to the conclusion: “Hey, logics is cool. I need to know more about it…!”

• Learn about **syntax** of logical languages
  – Construct valid terms and formulas from a given signature; detect invalid expressions

• Learn of **semantics** and **interpretation**
  – What are interpretations and why are they needed
    • Most importantly: syntax has no meaning without interpretation
  – Interpret given formulas and terms
2.0 Knowledge Generation

• Remember: Our long term goal is to build deductive DBs and expert systems...

• Basic question: **How can we generate new knowledge?**
  
  – Start with some knowledge that is (generally?) considered true (**axioms**)
  
  – Derive new knowledge in a consistent and understandable fashion… (**inference**)
  
  – Hmmm, …seems far from trivial
Inference comes in two major flavors

- **Inductive inference:**
  Perform multiple observations and draw a conclusion

- **Deductive inference:**
  Provide some true facts (axioms) and rules and then combine them to generate conclusions (theorems)
• Let’s do some time travel…

**sophism** (5\textsuperscript{th} century BC)

– Pre-socratic philosophy
– Only fragments survive
– Known through the writings of opponents like Plato or Aristotle

• **Rhetoric as a (paid) skill**

– Used for persuasion of others
– Use ambiguities of language in order to deceive or to support fallacious reasoning
First appearance of formal logics was around 330 BC in ‘Prior Analytics’ and ‘On Interpretation’ appearing in Aristotle’s Organon.

Logic was intended as a tool for valid philosophical arguments.

Aimed at formal and safe inference:
- Describing the process of deriving new knowledge from old knowledge or observations.
- Discovers many sophistic tricks and fallacies in ‘On Sophistical Refutations‘ in the Organon.
2.0 Logic as a Tool

• Sophistic Tricks and Fallacies:

  – Fallacy of Equivocation:
    • “A feather is light. What is light cannot be dark. Therefore, a feather cannot be dark.”

  – Fallacy of Composition:
    • “Each individual vote is rational. Therefore, combining multiple votes is also rational.”

  – Fallacy of Accident:
    • “Cutting people with knives is a crime. Surgeons cut people with knives. Surgeons are criminals.”

  – Fallacy of Begging the Question:
    • “Everybody thinks that the X900 is the hottest thing, because it is the hottest gadget right now.”
Propositional Logic deals with atomic logical statements and logical connectives in a merely structural sense

- **Atomic statements** cannot be further divided
  - Examples are ‘The earth is flat’ or ‘Socrates is dead’
- **Connectives** are ‘AND’, ‘OR’ and the implication ‘$\implies$’
- Basically all connectives are **truth functions** that evaluate to ‘true’ or ‘false’ in bivalent logic
  - There are also **multi-valued logics**, think for instance about NULL values in relational databases
• First approaches date back to Aristotle who discussed some basic principles in the collection ‘Metaphysics’ (around 4th century BC)
  – ‘A statement and its contradiction cannot be true at the same time’
  – ‘Every statement or its contradiction has to be true’
  – The technique of indirect proofs

• Propositional logic then has been heavily refined during medieval times
2.0 Propositional Logic

• A first sound and complete formalization for **truth values** was given by George Boole in 1847 with his algebraic calculus
  – Boolean Algebra

• Graphic representation by **Venn diagrams**

- x AND y
- x OR y
- NOT x
The first real calculus with implications was then formalized by Gottlob Frege (1879) and subsequently refined by Bertrand Russell (1910).

But propositional logic is the simplest kind of logical calculus...

- It does not investigate the statements themselves
- For instance quantifiers or predicates are not used, which limits the applications
- Sometimes referred to as zero\textsuperscript{th}-order logic
• For the special application in **deductive inference** from statements Aristotle introduced **term logic**
  – Term logics remained the dominating logical paradigm until the advent of **predicate logics** in the late 19th century

• Consists of three **basics constructs**
  – **Term**: A word representing ‘something’
  – **Proposition**: A combination of **two terms** (the subject and the predicate)
  – **Syllogism**: An inference where some proposition (conclusion) directly follows from two others (premises)
• **Terms**
  – A term per se is neither *true* nor *false*
  – Examples: Aristotle, man, mortal, blue, …

• **Propositions**
  – Provide a statement which is *either* true or false
  – Propositions have a **quantity** and a **quality**
    • Universal and affirmative: 'All men *are* mortal'
    • Existential and affirmative: 'Some men *are* philosophers'
    • Universal and negative: 'No man *is* immortal'
    • Existential and negative: 'Some men *are not* philosophers'
• The square of opposition defines the **allowed** logical conversions
The syllogism is the actual device of inference.

- The **minor premise** contains a minor term (subject) and a middle term (predicate).
- The **major premise** contains the same middle term (subject) and a major term (predicate).
- The **conclusion** contains the minor term as subject and the major term as predicate.
• In syllogisms of the four terms in the premises, one has to make the connection
  – Thus, one term has to appear twice and work as subject and predicate
• Based on the allowed conversations, we can define 256 different types of syllogisms
  – Only 24 are valid
• Examples
  – Universal-Affirmative UA-UA-UA
    • All Greeks are men. & All men are mortal.$
      \Rightarrow \text{All Greeks are mortal.}
2.0 Term Logic

– Universal-Affirmative UN-UA-UN

  - No reptiles have fur. & All snakes are reptiles.
    \[\Rightarrow\] No Snakes have fur.

– Universal-Affirmative PN-UA-PN

  - Some cats have no tail. & All cats are mammals.
    \[\Rightarrow\] Some cats have no tail.
• For Aristotele in propositions and syllogisms only **plurals** (universal terms) are possible
  – Term logic largely ignores **singular terms**
  – Can you say ‘Every **Socrates** is a philosopher’?
• Later, **singles** have been introduced predicking only one thing and treated as **universals**
  – All Socrates are men. & All men are mortals.
  \[\implies\text{All Socrates are mortals.}\]

• Introduced in the **Port-Royal-Logic** by Antoine Arnauld and Pierre Nicole (1662)

• Obviously, this is a little awkward...
• Eubulides of Miletus (4th century BC)
  – Philosopher of the Megarian School

• Heavily criticized Aristotle’s syllogisms
  – A grain of sand is no heap. Adding a single grain does not make a heap.
    ⇒ There is no heap of sand!
  – I still have, what I have not lost.
    I have not lost horns.
    ⇒ I have horns!
But some fallacies are not Aristotle’s fault

- For instance ‘Quaternio Terminorum’
- All adults love children. & All children love chocolate. ⇒ All adults love chocolate.

Where is the fallacy?

- All adults are children-lovers. All children are chocolate-lovers ⇒ …Nothing!!!
- Because only three terms are allowed in syllogisms
• Well, it does not seem easy to avoid all fallacies…
• The actual downfall of term logic was mainly due to **Gottlob Frege** (1879)
• Term logic dealt with **few logical constructs**
  – AND, OR, IF ... THEN..., NOT, SOME and ALL
• Frege recognized the need for **quantifiable variables and predicates** in mathematical statements
Before Frege a major problem was the distinction between **object** and **concept**

- Consider:
  
  'The *Morning Star* is *Venus*’ vs. ‘*Venus* is a *planet*’

- One sentence is reversible, the other is not… hence it cannot be the same ‘is’

  - What was needed is the concept of **objects** and **predicates** leading to **predicate logic**
  - The first ‘is’ means the equivalence of two **objects**, the second ‘is’ belongs to a binary predicate ‘is_a’ and in this case describes the **concept** of ‘being a planet’
• The work now was to **axiomatize** the new system of logic
  – Basic theory: mathematics is an extension of logic and therefore some (or all) mathematics is reducible to logic

• Foundation of **analytic philosophy**: logical clarification of thoughts can only be achieved by analysis of the logical form of philosophical propositions
  – **Neo-Positivism** with the Vienna Circle (Rudolf Carnap, Kurt Gödel, etc.) and the Berlin Circle (Hans Reichenbach, David Hilbert, etc.)
  – **Ideal language analysis** (e.g., Ludwig Wittgenstein)
• However, Frege’s mathematical logic for set theory still contained a contradiction

  – Russell’s paradox or Russell’s antinomy (1901) constructs a set containing exactly the sets that are not members of themselves

  – Imagine a barber shaving all people, if and only if they do not shave themselves…
  Does this barber shave himself?

  – Frege was frustrated and gave up…
The modern logical calculus was introduced by Bertrand Russell, 3rd Earl Russell

- Father of the **axiomatic set theory**
- Co-author of the ‘Principia Mathematica’ with Alfred North Whitehead
- **Idea:** if a complete and consistent set of axioms is known, every true theorem of the system can be derived eventually
- Taken up by **David Hilbert** to axiomatize **all** mathematics (Hilbert’s program)
The hope’s for complete axiomatization of all mathematics were shattered by Kurt Gödel’s incompleteness theorems.

– ‘On formally undecidable propositions of Principia Mathematica and related systems’ (1931)

1. If the system is consistent, it cannot be complete.
2. The consistency of the axioms cannot be proved within the system.
2.0 After Gödel

• Although it is not possible to formalize all mathematics…
  – It is possible to formalize essentially all the mathematics **that anyone uses**
  – First big success was by Kurt Gödel himself in 1929: the **completeness theorem for first order logic**
  – Any valid logical consequence of a series of axioms is provable by a finite deduction (the ‘formal proof’)

Knowledge-Based Systems and Deductive Databases – Wolf-Tilo Balke – IfIS – TU Braunschweig
2.1 Book Recommendation

• Tales of Logicians
  – How logics came to be...
  – “Logicomix: An Epic Search for Truth”
    • A. Doxiadis, C. Papadimitriou
    • Bloomsbury, 2009
  – Text Excerpt
    • Chapter 3 – “Bertrand Russell meets Gottlob Frege”
2.1 First-Order Logic

• The central idea behind first order logic (FOL) is to formally deduce from a set of facts which statements are true and which are false
  – Thus, we have to define what true and false is
  – In contrast to term logic and propositional logic, first order logic introduces the concept of predicates
    • Also called predicate logic of first order (1-PL)
    • A predicate can be used to group individual entities into types, i.e. 'Socrates is a man.' often written as predicate ‘man(Socrates)’
– Also, fine-grained quantification is possible
  • **Existential** quantification and **universal** quantification
  • e.g. 'All entities which are a man, are also mortal.'

• FOL is fully formalized
  – FOL **languages**
    • **Syntax**: How do valid statements look like?
  – FOL **interpretations**
    • **Semantic**: When is a statement true, when is it false?
  – FOL **systems**
    • **Deduction**: What statements can be deduced given a set of facts?
• Defining the language of first order logic roughly mimics **natural language**
  – Base building blocks are **formulas**, i.e. sentences containing true or false statements
  – A true or false statement can only be made when using a **predicate**
  – Predicates express something about some **terms**
    • 'Hector is a frog', '5 is greater than 3', '5 + 3 is even'
2.1 Syntax of First Order Logic

– Terms may represent either **objects** or **concepts** (and are thus **constant**)
  • 'Hector is a frog', '5 is uneven'
– Terms may be **variables** (and thus may represent a number of values)
  • 'something is a something else'
– Terms may use **functions** on other terms
  • '5+3 is even', 'The day after Monday is Tuesday'

• **Statements may be concatenated**
  – 'Hector is a frog and 5 is uneven'
  – 'If Hector is a frog, then 5 is uneven'
• When using a variable term in a statement in natural language, you may **assign** some value
  – ‘**something** tastes delicious’
    • Does not mean anything. What is ‘**something**’?
  – ‘A banana **tastes delicious**’
    • Now, ‘**something**’ is replaced (substituted) by just a single entity
Variables can be quantified

- **Universal-quantification:** for a statement to be considered true, the predicate has to be true **for all valid** substitutions of the variable
  - ∀ something (‘something smells nice’)

- **Particular-quantification:** for a statement to be considered true, the predicate has to be true **for at least one valid substitution** of the variable
  - ∃ something (‘something smells nice’)

Knowledge-Based Systems and Deductive Databases – Wolf-Tilo Balke – IfIS – TU Braunschweig
In the following, we will provide a construction mechanism for statements in a formal first order logic language.

A specific first order logic language can be defined as a quadruple $\mathcal{L} = (\Gamma, \Omega, \Pi, \chi)$.

- Attention: All elements of $\mathcal{L}$ are only symbols and have no meaning!
- In this section, we only discuss syntax!
• $\Gamma$ is the non-empty and decidable set of **constant symbols**
  – As constant symbols, we will usually use $a$, $b$, and $c$
    • If an entity of the real world is represented, we just use the entities name
  – **Constants may represent singular entities**, not types
  – Example:
    • $1, 2, 3, 4, \ldots$, but not natural numbers $\mathbb{N}$
    • ‘Hector, the frog’, ‘The Count’, ‘Tilo Balke’, but not ‘Frogs’, or ‘People‘
    • These are just character strings, not entities or objects!
2.1 First Order Logic Language

• \( \Pi := \bigcup_{n \in \mathbb{N}} \Pi_n \) is the disjunctive union of the finite sets \( \Pi_n \) of \( n \)-ary predicate symbols

  – As predicate symbols, we will usually use \( P, Q, \) and \( R \)
    • Also, when real word concepts are modeled, their respective name may be used

  – Predicates are used to define sets of elements, i.e. instead of listing all elements of a set, a predicate describes when a element belongs to a set and when not
    • A predicate \( P \) thus evaluates to either true or false
    • \( \{x | P(x)\} \) is the set of all elements \( x \) for which \( P \) is true

  – Example:
    • \( \text{Frog}(x) \) is symbol which might represent a predicate that evaluates to true for all \( x \) which are a frog
• X is the enumerable set of variables
  – Variables are usually denoted with $x$, $y$, and $z$
  – In first order logic, variables may only be used to represent a constant value
  – In some higher order logics (e.g. second order logic), you may also use variables to represent predicates or functions
    • A very powerful feature which induces a degree of complexity which we do not want to deal with…. 
• $\mathcal{L}$ is also called the **signature** of the language
  
  – For each application, you may define a specifically tailored signature

  – However, it used to be common (and still is within philosophy) to use a single **universal signature** for all application scenarios

    • $\Pi = \{P^1_1, P^1_2, P^1_3, \ldots, P^2_1, P^2_2, P^2_3, \ldots, \ldots\}$
    • $\Omega = \{f^1_1, f^1_2, f^1_3, \ldots, f^2_1, f^2_2, f^2_3, \ldots, \ldots\}$

    • Since this is not really convenient, we will avoid this…
2.1 Terms and Predicates

- **Terms** and **formulas** are the building blocks of logical languages.
- **Terms** are built from functions, variables, and constants.
  - However, they do not imply any true or false **statement**
    - “Hector”, “Day after tomorrow”, “5+7”
- Now, we have to insert terms into predicates to form a formula.
  - Formulas are statements which may be **true** or **false**
    - “Hector is a frog”, “Day after tomorrow is Wednesday”, “5+7 = 4”
    - **BUT**: Truth values are assigned later by the interpretation!
2.1 Terms

- The set of **terms** of a signature $T_{\mathcal{L}}$ is defined by the following rules
  - All **constant** symbols in $\Gamma$ are also terms
  - All **variable** symbols in $X$ are also terms
  - $f(t_1, \ldots, t_n)$ is a term iff $f \in \Omega_n$ is a $n$-ary **functional symbol** and $t_1, \ldots, t_n$ are terms
  - Nothing else is a term
  - If term does not contain any variables, it is called a **ground term**
• We define the set of **atomic formulas** of $\mathcal{L}$ $\mathcal{A}_\mathcal{L}$\n$=\{p(t_1, ..., t_n) | p \in \Pi_n \text{ and } t_1, ..., t_n \in T_\mathcal{L}\}$

– i.e. all defined **predicates** allocated with valid terms as arguments

– Formulas will later be interpreted to be **true** or **false**

– Atomic formulas thus state simple facts defined by predicates

  • e.g. 'Hector is a frog', 'Hector likes the Südsee'
2.1 Atomic Formulas

• Example
  – Constants $\Gamma = \{\text{Hector, Südsee, a, b, green, red, blue}\}$
  – Predicate Symbols $\Pi = \{\text{Frog}(x), \text{Lake}(x), \text{likes}(x, y), \text{hasColor}(x, y)\}$
  – Functional Symbols $\Omega = \{\text{colorOf}(x)\}$
  – Variables $X = \{x, y, z\}$

• Thus, the terms $T_{\mathcal{L}\{\Gamma, \Pi, \Omega, X\}}$ are
  $$\{\text{Hector}, \text{Südsee}, \text{a}, \text{b}, \text{green}, \text{red}, \text{blue}, x, y, z, \text{colorOf(Hector)}, \text{colorOf(Südsee)}, \ldots\}$$

• The set of atomic formulas $A_{\mathcal{L}\{\Gamma, \Pi, \Omega, X\}}$ is
  $$\{\text{Frog(Hector)}, \text{Frog(Südsee)}, \ldots, \text{Frog(z)}, \text{Frog(colorOf(Hector))}, \ldots, \text{Lake(Hector)}, \ldots\}$$
Furthermore, we can define formulas recursively by combining them with connectives

- All atomic formulas are formulas
- If \( W \) is a formula, then also \( \neg W \) is a formula
- If \( W_1 \) and \( W_2 \) are formulas, then also \( W_1 \land W_2 \), \( W_1 \lor W_2 \), \( W_1 \rightarrow W_2 \), \( W_1 \leftrightarrow W_2 \) are formulas
- If \( x \) is a variable and \( W \) a formula, then \( \forall x(W) \) and \( \exists x(W) \) are formulas
- Nothing else is a formula
2.1 Syntax Diagram

- **Constants \( \Gamma \)**
- **Variables \( X \)**
- **Term**
- **Predicates \( \Pi(\text{Terms}) \)**
- **Functions \( \Omega(\text{Terms}) \)**
- **Atomic formula**

**Formula**
- \( \vee \) Formula
- \( \wedge \) Formula
- \( \rightarrow \) Formula
- \( \leftrightarrow \) Formula
- \( \neg \) Formula
- \( \forall \) Formula
- \( \exists \) Formula
The logical connective symbols form the following precedence hierarchy (thus, parentheses may be avoided)

1. $\forall \exists$
2. $\neg$
3. $\land$
4. $\lor$
5. $\rightarrow \leftrightarrow$
Logical Language

- Given is a language $\mathcal{L} = (\Gamma, \Omega, \Pi, X)$ with
  - $\Gamma := \{a, b\}$
  - $\Omega := \{f(x), g(x, y)\}$
  - $\Pi := \{P, Q(x, y), R(x)\}$
  - $X := \{x, y\}$.
  - Which of the following strings are valid wrt. $\mathcal{L}$? Are they term, a formula, or an atom?
    - $f(g(x, y))$
    - $P$
    - $Q(x, y) \lor Q(a, b)$
    - $Q(g(f(a), x), f(y))$
    - $\forall a(R(a))$
    - $\exists x(f(x))$
    - $R(x) \rightarrow \neg R(x)$
    - $\neg R(\neg R(f(x)))$
Logical Language

- Given is a language \( \mathcal{L} = (\Gamma, \Omega, \Pi, X) \) with
  - \( \Gamma := \{a, b\} \)
  - \( \Omega := \{f(x), g(x, y)\} \)
  - \( \Pi := \{P, Q(x, y), R(x)\} \)
  - \( X := \{x, y\} \).
  - Which of the following strings are valid wrt. \( \mathcal{L} \)?
  - Are they a term, a formula, or an atom?
    - \( f(g(x, y)) \) – valid term
    - \( P \) – valid atom
    - \( Q(x, y) \lor Q(a, b) \) – valid formula
    - \( Q(g(f(a), x), R(y)) \) – invalid
    - \( \forall a(R(a)) \) – invalid
    - \( \exists x(f(x)) \) – valid formula
    - \( R(x) \rightarrow \neg R(x) \) – valid formula
    - \( \neg R(\neg R(f(x))) \) – invalid
2.1 Variable Bindings

• All variables which appear in a formula are free or bound

• For any formula $W$, three sets can be defined
  – $\text{vars}(W)$ containing all variables of $W$
  – $\text{free}(W)$ containing all free variables of $W$
  – $\text{bound}(W)$ containing all bound variables of $W$
  – $\text{vars}(W) = \text{free}(W) \cup \text{bound}(W)$
2.1 Variable Bindings

• Formulas can further be classified depending on their variables
  – Are variables bound or free (i.e. without specific value?)
  – Only quantifiers (∀, ∃) can bind variables

• For any formula \( W \), \( \text{free}(W) \) and \( \text{bound}(W) \) are recursively defined as following
  – \( \text{free}(W) := \text{vars}(W) \) and \( \text{bound}(W) := \emptyset \) if \( W \) is atomic
    • Atomic formulas have only free variables (if any)
  – \( \text{free}(¬W) := \text{free}(W) \) and \( \text{bound}(¬W) := \text{bound}(W) \)
    • Negation does not bind or unbind variables
2.1 Variable Bindings

- \( \text{free}(W_1 \land W_2) := \text{free}(W_1) \cup \text{free}(W_2) \) and \\
  \( \text{bound}(W_1 \land W_2) := \text{bound}(W_1) \cup \text{bound}(W_2) \)

  - Analogously for \( \lor \), \( \rightarrow \), and \( \leftrightarrow \)
  - **Binary connectives** merge the respective free and bound variable sets

- \( \text{free}(\forall x (W)) := \text{free}(W) \setminus \{x\} \) and \\
  \( \text{bound}(\forall x (W)) := \text{bound}(W) \cup \{x\} \)

  - Analogously for \( \exists \)
  - **Quantification** binds variables, e.g. the quantifier \( \forall \) binds the variable \( x \)
2.1 Variable Bindings

• Any formula $W$ with $\text{free}(W) = \emptyset$ is called closed, otherwise it is called open
  
  – Open formulas use all free variables as ‘parameters’
  
  • The truth value of open formulas depend on the value of free variables
  
  • Closed formulas do not depend on specific variable values and are thus of a constant truth value
    
    – Only depended on interpretation
2.1 Variable Bindings

• Examples

- \( F_1 \equiv P_1() \) is closed and free\((F_1) := \emptyset \)
- \( F_2 \equiv P_2(x_1, x_2) \) is open and free\((F_2) := \{x_1, x_2\} \)
- \( F_3 \equiv \forall x_1 (P_2(x_1, x_2)) \) is open and free\((F_3) := \{x_2\} \)
- \( F_4 \equiv \exists x_2 \forall x_1 (P_2(x_1, x_2)) \) is closed and free\((F_4) := \emptyset \)
- \( F_5 \equiv P_3(x_1) \land P_3(x_2) \) is open and free\((F_5) := \{x_1, x_2\} \)
- \( F_6 \equiv \text{Frog}(\text{Hector}) \) is closed and free\((F_6) := \emptyset \)
- \( F_7 \equiv \text{Frog}(x_1) \) is open and free\((F_7) := \{x_1\} \)
- \( F_8 \equiv \forall x, y (\text{Frog}(x) \land \text{Lake}(y) \rightarrow \text{likes}(x, y)) \) is closed
• Bound variables are only \textbf{valid within the scope of their respective quantifiers}

  – i.e. the same variable symbol might appear multiple time independently of each other

  – $F_1 \equiv \forall x_1 (P_3(x_1)) \land P_2(x_1)$ is open and \textit{free}(F_1):=\{x_1\} and \textit{bound}(F_1):=\{x_1\}

  • The first $x_1$ is \textbf{independent} of the other $x_1$!

  • As this is quite confusing, it is better to \textbf{rename} all occurrences of a bound variable symbol outside its scope
2.1 Rectification

• Any formula $W$ with $\text{free}(W) \cap \text{bound}(W) = \emptyset$ is called **rectified**
  
  – Thus, renaming occurrences of out-of-scope variables is called **rectification**
  
  – Example:
    
    • Bad: $F_1 \equiv \forall x_1 (P_3(x_1)) \land P_2(x_1)$
    
    • Rectified: $F_2 \equiv \forall x_1 (P_3(x_1)) \land P_2(x_2)$
      
      is open and $\text{free}(F_2) := \{x_2\}$ and $\text{bound}(F_2) := \{x_1\}$
  
  – From now on, we will only consider rectified formulas
2.1 Closures

- For any formula $W$ with $\text{free}(W)=\{x_1, x_2, ..., x_n\}$, a closure operation is defined
  - The closure operation binds all unbound variables
  - Universal closure: $\forall x_1, x_2, ..., x_n \ (W)$
  - Existential closure: $\exists x_1, x_2, ..., x_n \ (W)$
  - Example:
    - $W := \forall x_1 \ (P_3(x_1)) \land P_2(x_2)$
    - Existential closure of $W$: $\exists x_2 \forall x_1 \ (P_3(x_1)) \land P_2(x_2)$
• What is the difference between a term and an atom?
• What are open and closed formulas?
• What is rectification? Which problems does it address?
2.1 Syntax of FOL

Formula $W$ contains **connectives**?
- no: $W$ is **atomic**
- yes: $W$ is not atomic

**contains variables?**
- no: $W$ is a **ground formula**
- yes: $W$ is no ground formula

**contains free variables?**
- no: $W$ is closed
- yes: $W$ is open

**contains same variable with different scopes?**
- no: $W$ is rectified
- yes: $W$ is not rectified
A logical language $\mathcal{L}$ is given by the signature $\mathcal{L} = (\Gamma, \Omega, \Pi, \chi)$

- $\Gamma$ : All constant symbols
- $\Omega$ : All functional symbols
- $\Pi$ : All predicate symbols
- $\chi$ : All variable symbols

Attention: all elements of $\mathcal{L}$ are only symbols and have no meaning!
• There are recursive rules for combining function, constants, and variables to **terms**
  – Term: a yet un-interpreted expression which will later have some value

• There are recursive rules for combining predicates on terms to **formulas**
  – Formula: a yet un-interpreted **statement** which will later be either true or false

• Formulas can further be classified into atoms, open, closed, and rectified formulas
• The previous section just provides us with **syntactically correct** formulas
  
  – These are just **character strings** and have absolutely **no meaning**
  – All symbols occurring are really **just symbols** – nothing more
    
    • Of course, it would be a good idea if \( \land \) would mean ‘and’ and 5 would mean the natural number 5, but this is not mandatory

• In this section, we will provide formal ways to **interpret** a language \( \mathcal{L} = (\Gamma, \Omega, \Pi, \chi) \)
2.2 Semantics of First Order Logic

• Natural language analogy: Is the following statement true? 'No frog likes to eat flies'
  – Mhh... what is 'flies'? A small insects probably...
  – Same problem: What is a frog? Does it mean the famous amphibious truck?
  – 'Likes to eat'? Trucks don't eat, do they?
  – 'No frog...'? Might there be any frog who might like to eat flies? So, the truck doesn't... Maybe another frog?
  – Ok... Found one in the picture
    • At least the frog eats the fly, but does it like it?
• Even worse:
  – 'For all isomorphisms, their respective inverse is an homomorphism'
  – 'If tomorrow is Tuesday, then the day before yesterday is neither Saturday nor Sunday'

• We need to interpret those statements!
  – What do the words mean?
  – Which values can variables have? How do functions work? When are predicates true? What to do with quantifications like 'all' or 'any'?
  – How to evaluates concatenated statements?
2.2 Interpretation

• Example: Let be $\mathcal{L} = (\Gamma, \Omega, \Pi, X)$ with
  – $\Gamma := \{a, b\}$, $\Omega := \{f, g\}$, $\Pi := \{P\}$, $X := \{x, y, z\}$
  – Now, is $F \equiv P(f(a), g(b, x))$ true?

• Thus, we use an interpretation for capturing the semantics of our language $\mathcal{L}$
  – An interpretation assigns each term to an element of some universe of discourse
  – An interpretation assigns a truth value to each formula
    • i.e. decides which statements are true and which are false
• A universe of discourse is a non-empty set of objects, entities and concepts
  – Also sometimes referred to as domain of discourse or just universe or domain
  – Contains all entities and concepts related to our current application
    • e.g. represents a subset of the real world
2.2 Interpretation

• Formally, an interpretation is a quadrupel $I=(U, I_C, I_F, I_P)$
  
  – $U$ the universe of discourse
  
  • e.g. {Hector, Südsee, Addition, Humans, Root Function, …}
  
  • These are all real objects, entities or concepts, not just symbols or names

  – $I_C : \Gamma \rightarrow U$ is a mapping of all constant symbols to elements of the universe
  
  • e.g. Hector means the plastic frog on the coffee machine in IfIS,

  '5' means the natural number 5, etc
2.2 Interpretation

- \( I_F \) maps any \( f \in \Omega_n \) to an n-ary function, i.e.
  \[ I_F (f) : U \times \ldots \times U \rightarrow U \]
  n-ary function

  • e.g. the binary symbol '+' means additions of natural numbers, the unary symbol 'succ' means the natural successor function, etc

- \( I_P \) maps any \( p \in \Pi_n \) to an n-ary predicate, i.e.
  \[ I_P (p) \subseteq U \times \ldots \times U \]
  • e.g. the unary symbol 'Frog' represents the predicate deciding all frogs
2.2 Substitution

• Besides the interpretation of $\mathcal{L}$, there is a variable substitution $\rho: X \rightarrow U$
  – e.g. $x$ is ‘Hector the frog’, $y$ is ‘Südsee’

• Now, every term $t \in T_\mathcal{L}$ can be interpreted with respect to an interpretation $I_\mathcal{L}$ and an substitution $\rho$
  – For this, the term evaluation $I^*_\rho(t) =: t_1 \in U$ is used
  – $t_1$ is the result of the interpretation of $t$
2.2 Term Evaluation

• Term evaluation is intuitively quite simple
  – When a term is a constant, use the according constant interpretation.
  – If it is a variable, look up the variable substitution.
  – If it is a function, evaluate the function.

• Formally, following rules define the term evaluation $I^*_\rho(t)$
  – $t_I := I_C(c)$ if $t$ is a constant symbol, i.e. $t \in \Gamma$
  – $t_I := \rho(t)$ if $t$ is a variable symbol, i.e. $t \in \mathcal{X}$
  – $t_I := I_F(f)(I^*_\rho(t_1), \ldots, I^*_\rho(t_n))$
    if $t$ is a term of the form $f(t_1, \ldots, t_n)$
2.2 Term Evaluation

• Example again: Let be $\mathcal{L} = (\Gamma, \Omega, \Pi, X)$ with
  - $\Gamma := \{a, b\}$, $\Omega := \{f, g\}$, $\Pi := \{P\}$, $X := \{x, y, z\}$
• Now, is $F \equiv P(f(a), g(b, x))$ true?
• We need an interpretation $I = (U, I_C, I_F, I_P)$!
  - $U = \mathbb{N}$
  - $I_C : \Gamma \to U$, $\{a \mapsto 5, b \mapsto 3\}$
  - $I_F (f) : U \to U$, $n \mapsto n^2$
  - $I_F (g) : U \times U \to U$, $(n, m) \mapsto n + m$
  - $I_P (P) = \{(n, m) \in \mathbb{N}^2 \mid n < m\} \subseteq U \times U$
• $\Rightarrow 5^2 < 3 + x$
2.2 Formula Evaluation

• For interpreting formulas, we use a formula evaluation \( I_\rho : F_\mathcal{L} \rightarrow \{ \text{true, false} \} \)
  
  – The formula evaluation assigns a truth value \text{true} or \text{false} to each formula
  
  – An \textbf{atomic formula} (which consist of a single predicate \( P \)) is true if the predicate is fulfilled

1. \( I_\rho (W) := \{ \text{true} : \text{iff} I^*_\rho(t_1), ..., I^*_\rho(t_n) \in I_\rho(P), \text{false} : \text{otherwise} \}

  , if \( W \) is an \textbf{atomic formula} build by the n-ary predicate \( P \)
• In the previous definition, we implicitly used the closed world assumption
  – “Everything which is not explicitly mentioned in the universe does also not exist.”
  or
  “The universe enumerates all existing things.”
  – The opposite is called open world assumption
2.2 Formula Evaluation

• So, what happens if the open world assumption is used to evaluate atomic formulas?
  – Given is a language as follows:
    • $\Gamma$:={'Hector', 'Count'}, $\Omega$:={}, $\Pi$:={'Frog'}, $\chi$:={}
  – Given is an interpretation as follows:
    • $U = \{\text{Hector the frog, The Count}\}$
    • $I_C : \Gamma \to U$, {'Hector'}$\mapsto$Hector the frog, {'Count'}$\mapsto$The Count
    • $I_P (\text{Frog}) = \{(\text{Hector the frog})\} \subseteq U$
  – i.e. our universe contains Hector, the frog who is a frog and represented by the symbol ‘Hector’. Also, there is The Count who is not a frog, represented by ‘Count’.
2.2 Formula Evaluation

• Are the following atomic formulas true?
  – Frog(‘Hector’)  
    • TRUE, we know that Hector is a frog. Easy…
  – Frog(‘Count’)  
    • FALSE, also easy. We know that The Count is not a frog.
  – Frog(‘Tweety’)  
    • Uhmm….Who/what is Tweety??  
    • Closed World assumption: FALSE, Tweety cannot exist thus it is no frog.
    • Open World assumption: Don’t know, Tweety is not mentioned thus it might be a frog or not and the formula cannot be evaluated.

• Obviously, a system which most of the time answers with “I don’t know” is not that useful…
• With the interpretation $I_{\rho}$, the previous formula $F \equiv P(f(a), g(b, x))$ is interpreted as $5^2 < 3 + x$.

• Still, we cannot decide if the statement is true:
  – **Variable** substitution is missing!
  – For $\rho : X \rightarrow U$, $\{x \mapsto 1\}$, it is not true, $5^2 < 3 + 1$
    - $(25, 4) \notin \{(n, m) \in \mathbb{N}^2 | n < m\}$
  – For $\rho : X \rightarrow U$, $\{x \mapsto 99\}$, it is true, $5^2 < 3 + 99$
    - $(25, 102) \in \{(n, m) \in \mathbb{N}^2 | n < m\}$

• More interesting question: For which substitution $\rho$ is the $F$ true…?
  – Leads to logic programming
Now, we interpret non-atomic formula

- **Connectives** ¬, ∧, ∨, →, ↔ are interpreted as functions to \{true, false\} by using according proposition value function $I_\neg$, $I_\land$, $I_\lor$, $I_\rightarrow$, $I_\leftrightarrow$

<table>
<thead>
<tr>
<th>$W_1$</th>
<th>$W_2$</th>
<th>$I_\land(W_1, W_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$W_1$</th>
<th>$W_2$</th>
<th>$I_\lor(W_1, W_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$W_1$</th>
<th>$I_\neg(W_1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
2.2 Non-atomic Formulas

Thus, the formula evaluation $I^\rho: F_L \rightarrow \{\text{true, false}\}$ may be extended

- For any concatenation of formulas using connectives, the sub-formulas are evaluated and the value of the whole formula determined by the according value function.
2.2 Non-atomic Formulas

- If \( W, W_1, \) and \( W_2 \) are formulas, then

  2. \( I_\rho (\neg W) := I_\rho (\neg (I_\rho (W))) \)
  3. \( I_\rho (W_1 \land W_2) := I_\land (I_\rho (W_1), I_\rho (W_2)) \)
  4. \( I_\rho (W_1 \lor W_2) := I_\lor (I_\rho (W_1), I_\rho (W_2)) \)
  5. \( I_\rho (W_1 \rightarrow W_2) := I_\rightarrow (I_\rho (W_1), I_\rho (W_2)) \)
  6. \( I_\rho (W_1 \iff W_2) := I_\rightarrow (I_\rho (W_1), I_\rho (W_2)) \)

- Example:

\[
\neg P \land Q \lor R \rightarrow \neg S
\]

- [Diagram showing truth values for each proposition]

Knowledge-Based Systems and Deductive Databases – Wolf-Tilo Balke – IfIS – TU Braunschweig
Before interpreting quantified formulas, we will need the notation of modified substitutions

- i.e. we need to be able to modify some values of a given substitution

- Given is a substitution $\rho$ on the set $\{x_1, \ldots, x_n\} \subseteq X$ and the domain values $\{d_1, \ldots, d_n\} \subseteq U$
  
  Then the modified substitution $I_{\rho(x_1|d_1, \ldots, x_n|d_n)}(y)$ is:

  $\{d_i : \text{if } y \equiv x_i; \quad \rho(y) : \text{otherwise}\}$

- i.e. the variables of the substitution which are modified are explicitly listed, the others remain as they were before
• If variables are bound by an quantifier, the semantics are as following

– $\exists x(W)$: If there is any element of the universe for which the formula $W$ evaluates to true, the whole statement is true (and false otherwise)

$$I_\rho (\exists x(W)) := \{\text{true} : \text{if true} \in \{I_\rho(x|d) (W) \mid d \in U\} \}
\text{false} : \text{otherwise}$$

– Note, the we have to use the closed world assumption again! Without it, the statement cannot be evaluated!
• **Closed world** and quantification:
  – “Aliens do not exist”
  – **Closed World** Interpretation:
    • “We don’t know any aliens, i.e. any known thing is not an alien.”
    • “Thus, aliens do not exist.”
  – **Open World** Interpretation:
    • “We don’t know any aliens, i.e. any known thing is not an alien.”
    • “But otherwise, we know very little. Maybe there are aliens just around the corner on a Jupiter Moon?”
    • “I have absolutely no clue whether aliens exist or not…”
2.2 Quantified Formulas

- Universal quantification is treated analogously
  
  \[ \forall x(W) : \text{If there is any element of the universe for which the formula } W \text{ evaluates to } \text{false}, \text{the whole statement is } \text{false} \text{ (and true otherwise)} \]

  \[ I_\rho (\forall x(W)) := \begin{cases} 
  \text{true} : & \text{otherwise} \\
  \text{false} : & \text{if } \text{false} \in \{ I_\rho(x|d)(W) | d \in U \} 
  \end{cases} \]
• Discuss the relation between language, interpretation, and system

• What is the closed world assumption? Is it in general a good idea to use it? Why should a deductive DB use it?

• What is the difference between an interpretation and substitution?
  – Can they be merged? Would that be a good idea?
• **Next Lecture**
  – Logical Models
  – Horn Clauses
  – Logic Programming