University of Applied Sciences Northwestern Switzerland
School of Business

Master of Science
Business Information Systems

Semantics for Business Rules – Predicate Logic

Knut Hinkelmann

Semantic and Logical Foundations of Business Rules

- The SBVR initiative is intended to capture business facts and business rules
- Businss rules that may be expressed either informally or formally.
  - Formal statements of rules
    - may be transformed into logical formulations that
    - are used for exchange with other rule-based software tools.
  - Informal statements of rules may be exchanged as uninterpreted comments.
- Business rule expressions are classified as formal only if they are expressed purely using
  - fact types in the pre-declared schema for the business domain, and
  - logical/ mathematical operators, quantifiers, etc.

An Excursion into Logic

- The semantics of formal statements of business rules is defined by a mapping to predicate logic
- A predicate calculus consists of
  - a proof theory, made of
    - well-formed logical formulas
    - inference rules for deriving new formulas
  - a semantics, telling which interpretation of the symbols make the formulas true.

Predicate Logic vs Propositional Logic

- Propositional Logic is a formal system in which formulas
  - represent atomic propositions (having truth values true or false) or
  - are formed by combining propositions using logical connectives (and, or, not, …)
- Predicate Logic considers the deeper structure of propositions
  - Logical symbol: connectives, variables and quantifiers
  - Non-logical symbols: predicate and function symbols

Propositional Logic: Atomic Formulas

- In Propositional Logic, atomic formulas represent entire propositions (or sentences)
- Atomic formulas have a truth value: they are either true or false
- Examples:
  - It is raining
  - The street is wet
  - Paris is in France
  - The stars are blue
  - Fishes are sleeping in the trees
- Only the truth value is relevant, the structure or phrasing does not matter

Complex Formulas in Propositional Logic

- Complex formulas can be build using logical operators
- A and B are logical formulas having truth values

- Symbols denoting logical operators:
  - $\neg$ (negation – logical not)
  - $\wedge$ (conjunction – logical and)
  - $\vee$ (disjunction – logical or)
  - $\rightarrow$ (implication – logical condition)
  - $\leftrightarrow$ (equivalence – logical equivalence)

Truth values of formulas are computed using truth tables

A	B	$\neg$A	A $\wedge$ B	A $\vee$ B	A $\rightarrow$ B	A $\leftrightarrow$ B
true	true	false	true	true	true	true
true	false	false	false	true	false	false
false	true	true	false	true	true	false
false	false	true	false	false	true	true

Truth Values of Complex Statements

- Using the truth table it is also possible to derive the truth value of more complicated statements:
- What is the truth value of $A \wedge \neg B$ given that A and B are true?
- What is the truth value of
  - $A \wedge \neg A$
  - $A \vee \neg A$
  - $(A \wedge B) \vee \neg A \vee \neg B$
  - $((A \wedge B) \rightarrow C) \leftrightarrow (A \rightarrow (B \rightarrow C))$
- Exercise: Prove that $(A \rightarrow B)$ is equivalent to $(\neg A \vee B)$

Tautology and Contradiction

- A statement that is always true is called *logically true* or a *tautology*.
- A statement that is always false is called *logically false* or a *contradiction*.

Propositional Calculus

- A calculus consists of
  - a set of **axioms** (formulae representing the knowledge base)
  - a set of **infererence rules**: *syntactic* transformations which derive from a set of formulae a new formula
- Examples of Inference Rules for Propositional calculus:

Modus Ponens $$\frac{A,\ A \rightarrow B}{B}$$

Modus Tolens $$\frac{\neg B,\ A \rightarrow B}{\neg A}$$

Above the line are the formulas that are given and below the line is the new formula

Exercise: Deriving Truth Values using Propositional Calculus

- Represent the following statements in Propositional calculus
  - If it is raining then the street is wet
- You know that it is raining. Represent this fact so that you can apply an inference rule to derive new information
- What rule can be applied of you know that the street is not wet?

Limits of Propositional Logic

- Take the following example:

  A = All humans are mortal

  B = Socrates is a human

  C = Socrates is mortal

- It is intuitively valid that $\frac{A,\ B}{C}$

  (if A and B are true then C is also true)

- But there is no way in Propositional Logic to verify this argument, because the letters represent entire propositions and do not represent anything of the interal meaning of the sentences.

- Therefore we need a more powerfol logic which is called Predicate logic (or First Order Predicate Logic)

First-order Predicate Logic – Logical Symbols

The logical symbols include variables, logical operators and quantifiers.

- Variables are usually denoted by lowercase letters at the end of the alphabet *x*, *y*, *z*,..
- Symbols denoting quantifiers
  - $\forall$ (universal quantification, typically read as "for all")
  - $\exists$ (existential quantification, typically read as "there exists")
- Symbols denoting logical operators are usually denoted as
  - $\neg$ (negation – logical not)
  - $\wedge$ (conjunction – logical and)
  - $\vee$ (disjunction – logical or)
  - $\rightarrow$ (implication – logical condition)
  - $\leftrightarrow$ (equivalence – logical equivalence)

First-order Predicate Logic – Nonlogical Symbols

The nonlogical symbols include predicate symbols and function symbols

- The predicate symbols (or relation symbols) are often denoted by letters P, Q, R,... or p, q, r, s, t, ….
  - each predicate symbol has some arity ≥ 0
  - relations of arity 0 can be identified with propositional variables.
- The function symbols are often denoted by lowercase letters f, g, h,...
  - each predicate symbol has some arity ≥ 0
  - function symbols of arity 0 are called constant symbols, and are often denoted by lowercase letters at the beginning of the alphabet a, b, c,...

---

arity is the number of arguments

First-order Predicate Logic – Syntax of Terms

The set of terms is recursively defined by the following rules:

1. Any variable is a term,
2. Any constant symbol is a term,
3. If *f* is a function symbol of arity *n* and $t_1,\ldots,t_n$ *are terms, then* $f(t_1,\ldots,t_n)$ is a term
4. nothing else is a term

First-order Predicate Logic – Syntax of Formulas

The set of formulae is recursively defined by the following rules:

1. If P is a predicate symbol of arity n and $t_1,...,t_n$ are terms, then $P(t_1,...,t_n)$ is a formula (all these formulae are called atomic formula or atoms).
2. If A and B are formulae and the set of logical operators is $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$, then (A), $\neg A$, $A \wedge B$, $A \vee B$, $A \rightarrow B$, $A \leftrightarrow B$ are formulae.
3. If x is a variable, A is a formula and the set of quantifiers is $\{\forall, \exists\}$, then $\forall x\ A$ und $\exists x\ A$ are formulae.
4. Nothing else is a formula

In the formulae $\exists x\ A$ and $\forall x\ A$ the quantifiers bind the variable x. If a variable is not bound in a formula it is called a free variable.

First-order Predicate Logic – Interpretation

- To say whether a formula is true, we have to decide what the non-logical symbols mean.
- Interpretation: Let L be a language, i.e. the set of non-logical symbols. An interpretation consists of
  - a non-empty set D called the domain: The domain represents the set of things we are talking about
  - for each constant in L the assignment of an element in D
  - for each n-ary function symbol in L the assignment of a mapping from D^n to D
  - for each n-ary predicate symbol in L the assignment of a relation in D^n
    (this is equivalent to a mapping of D^n into {true, false})

First-order Predicate Logic – Interpretation

- Meaning of logic operators
  - The truth values of the logical operators are defined by the some truth tables as propositional logic

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
true	true	false	true	true	true	true
true	false	false	false	true	false	false
false	true	true	false	true	true	false
false	false	true	false	false	true	true

- In addition, the quantifiers have to following meaning
  - The formula $\exists x\, F$ is true in the interpretation, if there is an assignment of x with an individual such that F is true
  - The formula $\forall x\, F$ is true in an interpretation, if for every assignment of x the formula F

Exercise

- Find a domain and two unary predicates P(x) and Q(x) such that

  $$\forall x\, (P(x) \rightarrow Q(x))$$

- Find a domain and two unary predicates P(x) and Q(x) such that

  $\exists x\, (P(x) \land Q(x))$ is false

  and $(\exists x\, P(x)) \land (\exists x\, Q(x))$ is true

- Find a domain and a binary predicate P(x,y) such that

  $\exists x\, (P(x,a) \land P(x,b))$ is false

  and $(\exists x\, (P(x,b)) \land (\exists x\, P(x,b))$ is true

Translating from Natural Language into Predicate Logic

- When translating arguments from Natural Language into Predicate Logic the following rules should be followed:
  - Universal Conditional: All As are Bs

    $\forall x\ (A(x) \rightarrow B(x))$
  - Existentially qualified condition: Some As are Bs

    $\exists x\ (A(x) \wedge B(x))$
- It is common to make the following mistakes:
  - All As are Bs: $\forall x\ (A(x) \wedge B(x))$
    „For all x, x are A and x are B" (-> **wrong!**)
  - Some As are Bs: $\exists x\ (A(x) \rightarrow B(x))$
    „For some x, if x is an A then x is a B" (-> **wrong!**)

The Meaning of Quantifiers

- Consider the sentence „All unicorns have one horn“
- This can be translated as

  $\forall x\ (\text{Unicorn}(x) \rightarrow \text{Horn}(x))$

  (or equivalent: $\forall x\ (U(x) \rightarrow H(x))$)

  The meaning is: „If something is a unicorn then it has one horn“.
  It does not imply that unicorns exist
- Similar, the expression „not all men are brave“ can be translated as

  $\neg\ \forall x\ (M(x) \rightarrow B(x))$

  It does not imply that there exist any men or that there is a man who is brave.
- Universal quantification makes no claims about existence. Therefore we need the existential quantifier

First-order Predicate Logic – Calculus

- A calculus consists of
  - a set of **axioms** (formulae representing the knowledge base)
  - a set of **infererence rules**: *syntactic* transformations which derive from a set of formulae a new formula
- Examples of Inference Rules:

Modus Ponens $\dfrac{A,\ A \rightarrow B}{B}$

Substitution $\dfrac{\forall x\ P}{P\ \{x/a\}}$

Modus Tolens $\dfrac{\neg B,\ A \rightarrow B}{\neg A}$

Modus Ponens with substitution $\dfrac{P\ (a),\ \forall x{:}\ (P(x) \rightarrow Q(x))}{Q\ (a)}$

Negation with Quantifiers

- The expression „There are no unicorns“ can be represented in two ways

	$\forall x\, (\neg U(x))$	an universal negation
or	$\neg \exists x\, (U(x))$	a negated existential

- Some transformations that convert between universal and existential quantification:

♦ Something is A	$\exists x\, (A(x))$	$\neg \forall x\, (\neg A(x))$
♦ Nothing is A	$\neg\exists x\, (A(x))$	$\forall x\, (\neg A(x))$
♦ Everything is A	$\forall x\, (A(x))$	$\neg\exists x\, (\neg A(x))$
♦ Some As are Bs	$\exists x\, (A(x) \wedge B(x))$	$\neg \forall x\, (\neg (A(x) \wedge B(x)))$
♦ All As are Bs	$\forall x\, (A(x) \rightarrow B(x))$	$\neg\exists x\, (A(x) \wedge \neg B(x))$
♦ No As are Bs	$\neg\exists x\, (A(x) \wedge B(x))$	$\forall x\, (A(x) \rightarrow \neg B(x))$

First-order Predicate Logic – Models and Consequences

- An interpretation that makes a formula F true is called a model of F
- Logical consequence:
  - A formula G is a logical consequence of a formula F, if all models of F are also models of G. This is written as

$$F \models G$$

First-order Predicate Logic – Calculus

- If a formula F can be derived from a set of formulas $F_1,\dots,F_n$ by a sequence of inference rule applications, we write

$$F_1, \dots, F_n \vdash F$$

(One says that there is a proof for F from $F_1,\dots,F_n$)

- The control system that selects and applies the inference rules is called **inference procedure**.
- An inference procedure should be
  - **correct**, i.e. if $F_1, \dots, F_n \vdash F$ then $F_1, \dots, F_n \models F$
  - **complete**, i.e. if $F_1, \dots, F_n \models F$ then $F_1, \dots, F_n \vdash F$