UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

Özgür L. Özçep

Logic, Logic, and Logic

Lecture 1: Motivation and Overview

19 October, 2016

Foundations of Ontologies and Databases for Information Systems

CS5130 (Winter 16/17)

Organizational Stuff

Organization

- Lectures with integrated exercises (sometimes homework)
- Exercise slot may vary: so come to the lectures
- **Start**: Today, Wed, 19 October, 2016, 16.05h
- Lecture and exercise related material in Moodle "Grundlagen von Ontologien und Datenbanken für Informationssysteme - CS5130"
- **Oral exam** at the end of the semester
  - **Register** for the course in Moodle
  - Prerequisite for exam: At least 50 percent of exercises solved successfully
- The lectures and the exercises are in English

Organization

- Lectures with integrated exercises (sometimes homework)
- Exercise slot may vary: so come to the lectures
- **Start**: Today, Wed, 19 October, 2016, 16.05h
- Lecture and exercise related material in Moodle "Grundlagen von Ontologien und Datenbanken für Informationssysteme - CS5130"
- **Oral exam** at the end of the semester
  - **Register** for the course in Moodle
  - Prerequisite for exam: At least 50 percent of exercises solved successfully
- The lectures and the exercises are in English

Sometimes English Becomes Less Important

Prologue

La loi 101 (Charte de la langue française)

Principe du deux pour un□le texte français doit être écrit en caractères deux fois plus gros que ceux de la version en langue étrangère.

Two for one principle□an english (for clarity) text should be written in characters twice smaller than its french counterpart.

Exception□ the english version of the text of the Law itself can be written in characters five times bigger than the french original.

Slide example by Bruno Poizat from a conference talk

- Model Theorist
- Has a wonderful (unconventional) book on model theory
  - Was not well received (for some years)
  - until he translated it into English

Lit: B. Poizat. A Course in Model Theory. Universitext. Springer Verlag, 2000.

Sometimes English Becomes Less Important

Prologue

La loi 101 (Charte de la langue française)

Principe du deux pour un□le texte français doit être écrit en caractères deux fois plus gros que ceux de la version en langue étrangère.

Two for one principle□an english (for clarity) text should be written in characters twice smaller than its french counterpart.

Exception□the english version of the text of the Law itself can be written in characters five times bigger than the french original.

Slide example by Bruno Poizat from a conference talk

- Model Theorist
- Has a wonderful (unconventional) book on model theory
  - Was not well received (for some years)
  - until he translated it into English

Lit: B. Poizat. A Course in Model Theory. Universitext. Springer Verlag, 2000.

Plan

- Logic, Logic, Logic (2 lectures)
- Logical Foundations of Database Systems: Finite Model Theory (2 lectures)
- Semantic Integration with OBDA: Bridging the DB and Ontology World (2-3 lectures)
- Semantic Integration on Ontology Level: Ontology Integration (2-3 lectures)
- Stream Processing (2-3 lectures)
- Process Analysis and Design (2-3 lectures)

First-Order Logic

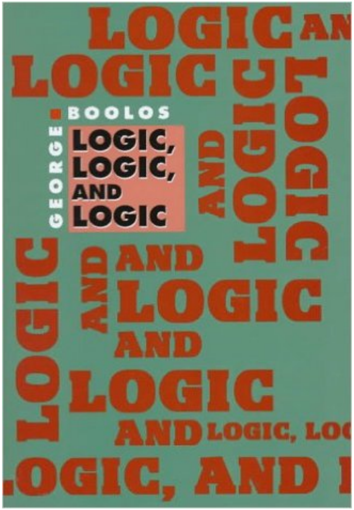
"Logic, Logic, and Logic"

- Interesting collection of essays
- Rather "philosophical logic"

- But we adopt the motto:

  Logic everywhere !

- We are interested not only in logics per se but
- (Knowledge on) logics useful for computer science



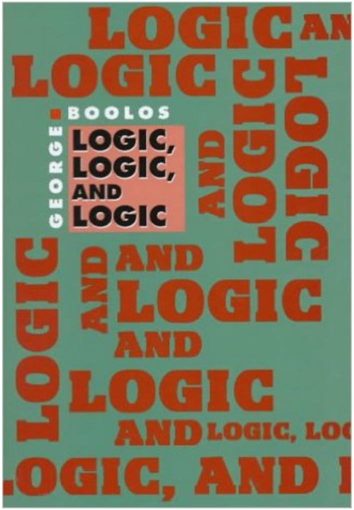
"Logic, Logic, and Logic"

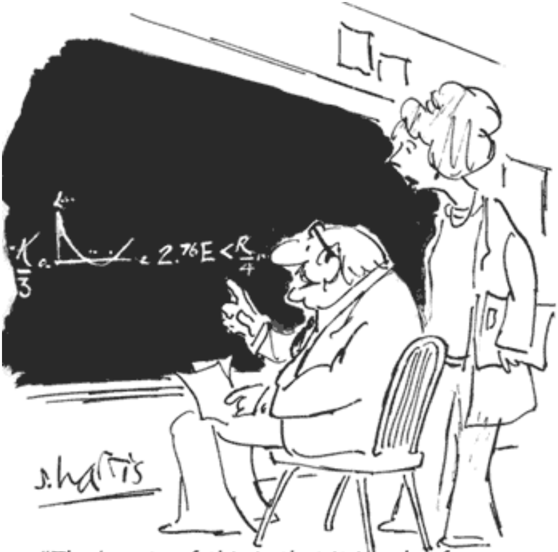
- Interesting collection of essays
- Rather "philosophical logic"

- But we adopt the motto:

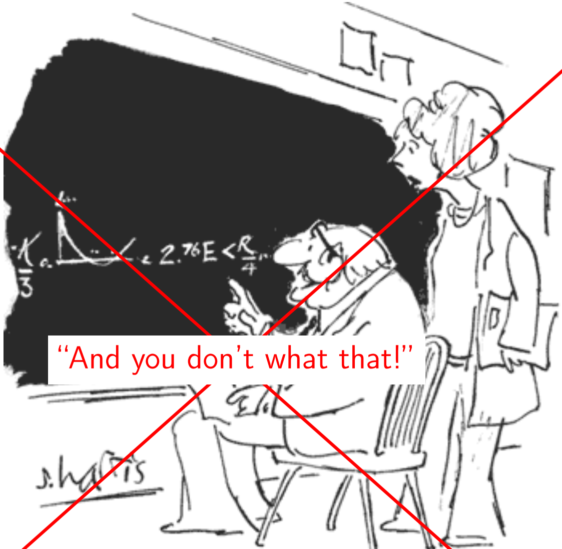
  Logic everywhere !

- We are interested not only in logics per se but
- (Knowledge on) logics useful for computer science



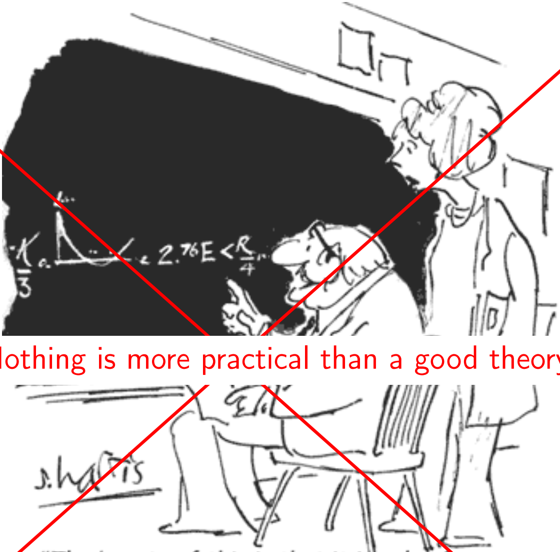


"The beauty of this is that it is only of theoretical importance, and there is no way it can be of any practical use whatsoever."



"The beauty of this is that it is only of theoretical importance, and there is no way it can be of any practical use whatsoever."

“And you don’t what that!”



But: "Nothing is more practical than a good theory"

"The beauty of this is that it is only of theoretical importance, and there is no way it can be of any practical use whatsoever."

Logic and Logics

- Science of logic
  - investigates mathematical **structures** (static and dynamic)
  - and formal **languages** to describe them
  - distinguishing between **syntax**
  - and **semantics** (truth conditions for sentences)
  - providing notions of **satisfaction, entailment** (from semantics)
  - and of **provability, inference (calculus)**

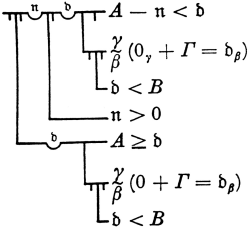
- A logic: A language with syntax, semantics (and possibly calculus)

- There are many different logics (within computer science)
- But in any case somehow related to first-order logic

First-Order Logic (FOL)

- Also called predicate logic (or quantification logic)
- Aristotelian syllogisms already incorporate restricted FOL
  - All Philosophers are wise men. All wise men are nice. Hence all Philosophers are nice men.
  - Restricted to unary predicates

- Modern FOL started with Frege's "Begriffsschrift"
  - language constructs based on constants, variables, predicates, functions, boolean connectives, quantifiers
  - Formal axioms and inference rules
  - His 2-dimensional representation format aesthetic but not practical



FOL Structures

- A formalism to investigate (mathematical) **structures**

$$\mathfrak{A} = (A, R_1^{\mathfrak{A}}, \ldots R_n^{\mathfrak{A}}, f_1^{\mathfrak{A}}, \ldots, f_m^{\mathfrak{A}}, c_1^{\mathfrak{A}}, \ldots, c_l^{\mathfrak{A}})$$

- **(Non-logical) Vocabulary**
  - Relation symbols/predicates R_i with arities
  - Function symbols f_i (with arities)
  - Constant symbols c_i

- Components of the structure
  - **Universe/Domain** A
  - **Interpretations/denotations** of nonlogical symbols
    - Relation $R^{\mathfrak{A}} \subseteq A^n$ (for n-ary relation symbol R)
    - Function $f^{\mathfrak{A}} \in A^n \longrightarrow A$ (for n-ary function symbol f)
    - Individuals $c^{\mathfrak{A}} \in A$ (for constants c)

Example FOL Structures

- **Graphs** $\mathfrak{G} = (V, E^{\mathfrak{G}})$
  1. V = nodes of the graph
  2. $E^{\mathfrak{G}} \subseteq V^2$ = edges of the graph

- **Undirected, loopless graphs** $\mathfrak{G} = (V, E^{\mathfrak{G}})$
  1. as above
  2. as above
  3. Additionally: edge relation is symmetric and a-reflexive

- Need an appropriate language to formulate constraints such as in 3.

FOL Syntax

- Allow variables $(x_1, x_2, \ldots)$ and logical constructors

- **Terms**
  - variables and constants are terms
  - if $t_1, \ldots, t_n$ are terms, so is $f(t_1, \ldots, t_n)$ (for n-ary function symbol f

- **Formulae**
  - $t_i = t_j$ and $R(t_1, \ldots, t_n)$ (for terms t_i and n-ary relation) R
  - If ϕ is a formula, so are
    - $\neg\phi$ ("Not ϕ")
    - $\forall x\ \phi$ ("For all x it holds that ϕ")
    - $\exists x\ \phi$ ("There is an x s.t. ϕ")

  - If ϕ, ψ are formula, so are
    - $(\phi \wedge \psi)$ ("ϕ and ψ")
    - $(\phi \vee \psi)$ ("ϕ or ψ")
    - $(\phi \rightarrow \psi)$ ("If ϕ then ψ")
    - $(\phi \leftrightarrow \psi)$ ("ϕ iff ψ")

FOL Syntax

- Allow variables ($x_1, x_2, \dots$) and logical constructors

- **Terms**
  - variables and constants are terms
  - if $t_1, \dots, t_n$ are terms, so is $f(t_1, \dots, t_n)$ (for n-ary function symbol f

- **Formulae**
  - $t_i = t_j$ and $R(t_1, \dots, t_n)$ (for terms t_i and n-ary relation) R
  - If ϕ is a formula, so are
    - $\neg\phi$ ("Not ϕ")
    - $\forall x\ \phi$ ("For all x it holds that ϕ")
    - $\exists x\ \phi$ ("There is an x s.t. ϕ")

  - If ϕ, ψ are formula, so are
    - $(\phi \wedge \psi)$ ("ϕ and ψ")
    - $(\phi \vee \psi)$ ("ϕ or ψ")
    - $(\phi \rightarrow \psi)$ ("If ϕ then ψ")
    - $(\phi \leftrightarrow \psi)$ ("ϕ iff ψ")

FOL Semantics

- **Interpretation** $\mathcal{I} = (\mathfrak{A}, \nu)$
  - ν assigns to all variables elements from domain A
  - Needed to deal with open formulae
    e.g. $\forall y\; R(y, x)$ open/free in variable x

- x-**Variant** $\mathcal{I}_{[x/d]}$
  same as $\mathcal{I}$ but with $d \in A$ assigned to x

- **Interpretation of terms**
  - $\mathcal{I}(c) = c^{\mathfrak{A}}$
  - $\mathcal{I}(x) = \nu(x)$
  - $\mathcal{I}(f(t_1, \ldots, t_n)) = f^{\mathfrak{A}}(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n))$

Because dealing with variables is non-trivial...

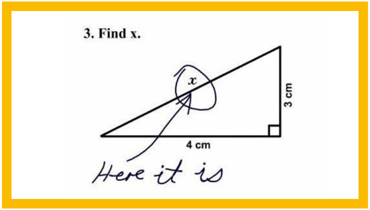
FOL Semantics

- **Interpretation** $\mathcal{I} = (\mathfrak{A}, \nu)$
  - ν assigns to all variables elements from domain A
  - Needed to deal with open formulae
    e.g. $\forall y\ R(y, x)$ open/free in variable x

- x-**Variant** $\mathcal{I}_{[x/d]}$
  same as $\mathcal{I}$ but with $d \in A$ assigned to x

- **Interpretation of terms**
  - $\mathcal{I}(c) = c^{\mathfrak{A}}$
  - $\mathcal{I}(x) = \nu(x)$
  - $\mathcal{I}(f(t_1, \ldots, t_n)) = f^{\mathfrak{A}}(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n))$

Because dealing with variables is non-trivial...



FOL Semantics

- **Satisfaction relation $\models$**
  - $\mathcal{I} \models t_1 = t_2$ iff $\mathcal{I}(t_1) = \mathcal{I}(t_2)$
  - $\mathcal{I} \models R(t_1, \ldots, t_n)$ iff $(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n)) \in R^{\mathfrak{A}}$

  - $\mathcal{I} \models \neg\phi$ iff not $\mathcal{I} \models \phi$

  - $\mathcal{I} \models (\phi \wedge \psi)$ iff $\mathcal{I} \models \phi$ and $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \vee \psi)$ iff $\mathcal{I} \models \phi$ or $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \rightarrow \psi)$ iff: If $\mathcal{I} \models \phi$ then $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \leftrightarrow \psi)$ iff: $\mathcal{I} \models \phi$ iff $\mathcal{I} \models \psi$

  - $\mathcal{I} \models \forall x\ \phi$ iff: For all $d \in A$: $\mathcal{I}_{[x/d]} \models \phi$
  - $\mathcal{I} \models \exists x\ \phi$ iff: There is $d \in A$ s.t. $\mathcal{I}_{[x/d]} \models \phi$

- Known result: ν can be assumed to be defined only for the free variables in the formula.
- Terminology $\mathcal{I}$ satisfies ϕ, $\mathcal{I}$ makes ϕ true, $\mathcal{I}$ is a model for ϕ
- We also write $\mathfrak{A} \models \phi(\vec{x}/\nu)$

FOL Semantics

- **Satisfaction relation** $\models$
  - $\mathcal{I} \models t_1 = t_2$ iff $\mathcal{I}(t_1) = \mathcal{I}(t_2)$
  - $\mathcal{I} \models R(t_1, \ldots, t_n)$ iff $(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n)) \in R^{\mathfrak{A}}$

  - $\mathcal{I} \models \neg\phi$ iff not $\mathcal{I} \models \phi$

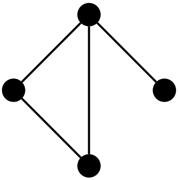
  - $\mathcal{I} \models (\phi \wedge \psi)$ iff $\mathcal{I} \models \phi$ and $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \vee \psi)$ iff $\mathcal{I} \models \phi$ or $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \rightarrow \psi)$ iff: If $\mathcal{I} \models \phi$ then $\mathcal{I} \models \psi$
  - $\mathcal{I} \models (\phi \leftrightarrow \psi)$ iff: $\mathcal{I} \models \phi$ iff $\mathcal{I} \models \psi$

  - $\mathcal{I} \models \forall x\ \phi$ iff: For all $d \in A$: $\mathcal{I}_{[x/d]} \models \phi$
  - $\mathcal{I} \models \exists x\ \phi$ iff: There is $d \in A$ s.t. $\mathcal{I}_{[x/d]} \models \phi$

- Known result: ν can be assumed to be defined only for the free variables in the formula.
- Terminology $\mathcal{I}$ satisfies ϕ, $\mathcal{I}$ makes ϕ true, $\mathcal{I}$ is a model for ϕ
- We also write $\mathfrak{A} \models \phi(\vec{x}/\nu)$

Examples

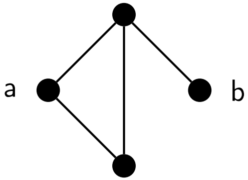
- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \, E(x, y)$ $\quad\quad \mathfrak{G} \models \phi_1?$
- $\phi_2(x) := \exists y \, \exists z \, E(x, y) \wedge E(x, z) \wedge E(y, z)$ $\quad\quad \mathfrak{G} \models \phi_2(x/a)$
- $\phi_3(x, y) := E(x, y)$ $\quad\quad \mathfrak{G} \models \phi_3(x/a, y/b)$

Examples

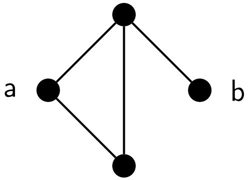
- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \, E(x, y)$ $\qquad \mathfrak{G} \models \phi_1$ Yes!
- $\phi_2(x) := \exists y \, \exists z \, E(x, y) \wedge E(x, z) \wedge E(y, z)$ $\qquad \mathfrak{G} \models \phi_2(x/a)$
- $\phi_3(x, y) := E(x, y)$ $\qquad \mathfrak{G} \models \phi_3(x/a, y/b)$

Examples

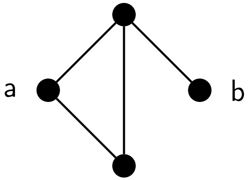
- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \; E(x, y)$ $\qquad \mathfrak{G} \models \phi_1$ Yes!
- $\phi_2(x) := \exists y \, \exists z \; E(x, y) \land E(x, z) \land E(y, z)$ $\qquad \mathfrak{G} \models \phi_2(x/a)$?
- $\phi_3(x, y) := E(x, y)$ $\qquad \mathfrak{G} \models \phi_3(x/a, y/b)$

Examples

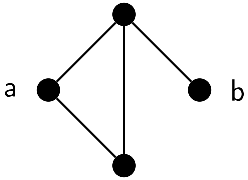
- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \; E(x, y)$ $\quad \mathfrak{G} \models \phi_1$ Yes!
- $\phi_2(x) := \exists y \, \exists z \; E(x, y) \wedge E(x, z) \wedge E(y, z)$ $\quad \mathfrak{G} \models \phi_2(x/a)$ Yes!
- $\phi_3(x, y) := E(x, y)$ $\quad \mathfrak{G} \models \phi_3(x/a, y/b)$

Examples

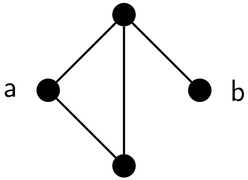
- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \; E(x, y)$ $\quad \mathfrak{G} \models \phi_1$ Yes!
- $\phi_2(x) := \exists y \, \exists z \; E(x, y) \wedge E(x, z) \wedge E(y, z)$ $\quad \mathfrak{G} \models \phi_2(x/a)$ Yes!
- $\phi_3(x, y) := E(x, y)$ $\quad \mathfrak{G} \models \phi_3(x/a, y/b)$?

Examples

- Consider loopless, symmetric graphs $\mathfrak{G} = (G, E^{\mathfrak{G}})$



- $\phi_1 := \exists x \, \exists y \; E(x, y)$ $\qquad \mathfrak{G} \models \phi_1$ Yes!
- $\phi_2(x) := \exists y \, \exists z \; E(x, y) \wedge E(x, z) \wedge E(y, z)$ $\qquad \mathfrak{G} \models \phi_2(x/a)$ Yes!
- $\phi_3(x, y) := E(x, y)$ $\qquad \mathfrak{G} \models \phi_3(x/a, y/b)$ NO!

Entailment

- $X \models \phi$ iff all models of X are models of ϕ
  - We say: X **entails** ϕ or ϕ follows from X
  - X: set of sentences
  - ϕ: sentence

- Note: entailment definition (per se) not easy implementable $\Longrightarrow$: Notion of derivibility/inference in a calculus (see later lectures)

Algorithmic Problems in First-Order Logic

- **Model Checking**:
  - Input: graph (or generally structure) $\mathfrak{G}$, formula $\phi(x_1, \ldots, x_n)$ and assignment $[x_1/a_1, \ldots, x_n/a_n]$
  - Output: Is $\mathfrak{G} \models \phi(x_1/a_1, \ldots, x_n/a_n)$ the case?

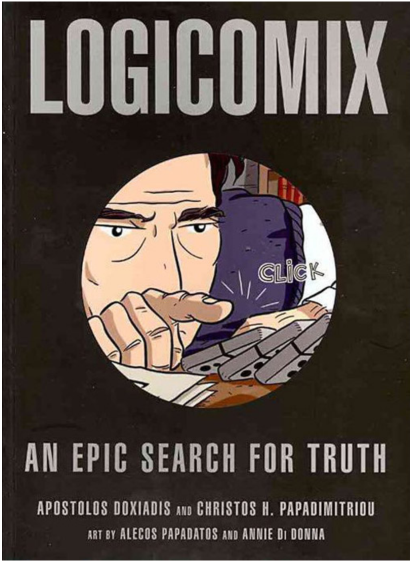
- **Satisfiability Problem**
  - Input: sentence ϕ
  - Output: Does there exist a structure $\mathfrak{G}$ s.t. $\mathfrak{G} \models \phi$?

- **Complexity of problems**
  - Model checking problem is decidable and PSPACE complete (in combined complexity)
  - Satisfiability is undecidable

Role of Logic for/in Computer Science

The Burden of Logic in the 19-20th Century

- Role of logic as a foundation for all of mathematics
- Literature hint: Logicomix
  - fantastic graphic novel
  - Narrator: Philosopher and logician B. Russell
  - About the illusions, disillusions, and landmarking results at the end of the 19th century



Foundations of Mathematics with Mathematical Logic

- Attempts to find formal foundation for mathematical logic
- Hilberts Program (1900-1928)
  - Mathematics is consistent
  - Mathematics is (semantically) complete
  - Mathematics is decidable

Awakening

- Young Gödel proves (1931-33)
  - arithmetics not complete
  - consistency of set theory not provable

- Church/Turing (1936/37)
  - First-order logic is not decidable
  - Valid sentences not recursive
  - Sentences true in arithmetic not recursively enumerable (semi-decidable)

- Nonetheless there are the following positive insights
  - Syntactically completeness for FOL (Gödel, 1930)
  - ZFC (Zermelo-Fraenkel Set Theory) can be used to formalize all contemporary mathematics

Awakening

- Young Gödel proves (1931-33)
  - arithmetics not complete
  - consistency of set theory not provable
- Church/Turing (1936/37)
  - First-order logic is not decidable
  - Valid sentences not recursive
  - Sentences true in arithmetic not recursively enumerable (semi-decidable)
- Nonetheless there are the following positive insights
  - Syntactically completeness for FOL (Gödel, 1930)
  - ZFC (Zermelo-Fraenkel Set Theory) can be used to formalize all contemporary mathematics

The Unusual Effectiveness of Logic

- Logic (Research) and Computer Science had fruitful effects onto each other
- Logic even more w.r.t. CS (than w.r.t. mathematics)
- "Logic is the calculus of CS"

Lit: M. Y. Vardi. From philosophical to industrial logics. In Proceedings of the 3rd Indian Conference on Logic and Its Applications, ICLA'09, pages 89–115, Berlin, Heidelberg, 2009. Springer-Verlag.

Lit: J. Y. Halpern, R. Harper, N. Immerman, P. G. Kolaitis, M. Y. Vardi, and V. Vianu. On the unusual effectiveness of logic in computer science. Bull. Symbolic Logic, 7(2):213–236, 2001.

Why is this the Case?

- Logic is so general that it allows to
  - talk precisely about the objects within a computer/computation model
  - specify and reason about the properties of runs in the model
- Even more: One can characterize complexity classes with logics (Descriptive Complexity)

So ...

As an upcoming computer scientist (in academia or industry) you should train in formal models, in particular **logics**, because:

- you want to **apply successfully** for a job
- But more importantly: you want to **keep** your job

Computer Science Areas Effected by Logic Research

- Database Systems
- Ontology-Based Information Systems
- Semantic Integration
- Computer-Aided Verification (Model Checking)
- Computational Complexity
- High-Level Stream Processing
- Multi-Agent Systems
- Machine Learning (e.g. probabilistic graph models and logics)
- Semantic Web
- Logic Programming
- Knowledge Representation
- Semantics of Programms
- Digital Design ...

Computer Science Areas Effected by Logic Research

- Database Systems
- Ontology-Based Information Systems
- Semantic Integration
- Computer-Aided Verification (Model Checking)
- Computational Complexity
- High-Level Stream Processing
- Multi-Agent Systems
- Machine Learning (e.g. probabilistic graph models and logics)
- Semantic Web
- Logic Programming
- Knowledge Representation
- Semantics of Programms
- Digital Design ...

This course

Computer Science Areas Effected by Logic Research

- Database Systems
- Ontology-Based Information Systems
- Semantic Integration
- Computer-Aided Verification (Model Checking)
- Computational Complexity
- High-Level Stream Processing
- Multi-Agent Systems
- Machine Learning (e.g. probabilistic graph models and logics)
- Semantic Web
- Logic Programming
- Knowledge Representation
- Semantics of Programms
- Digital Design ...

This course
Other courses of module "Web and Data Science" (CS4513)
This semester: "Web-Mining-Agenten"

Effects of Computer Science to Logic Research

- Focus/Intensive research on finite structures
  - Objects of computation are finite (Finite Model Theory)
  - But: potentially infinite structures (such as infinite DBs or streams) are useful as well
- Need for extensions of FOL
  - Higher-order logics (quantification over sets/relations)
  - Recursion (Datalog)
- Feasibility of reasoning services $\Longrightarrow$ restrictions of FOL
  - Modal and temporal logics
  - Description Logics
- Connections of logic and automata models
  - Regular expressions, finite automata, sequential logics
  - Buechi automata
- Logic engineering
- Different forms of inference ...

Overview on Course With Examples

Example: Logic in DB Research (Lectures 3-4)

- Travel DB with direct connection flights
- Reachability query
- SQL allows for recursion (CONNECT key word)
- But is it really necessary?

Table Flight

Start	End
Hamburg	Berlin
Hamburg	New York
New York	Berlin
...	...

Query Q_{reach}: List all cities reachable from Hamburg!

Intuitively without recursion:

$$
\begin{aligned}
Q_{reach}(x) \quad = \quad & Flight(Hamburg, x) \vee \\
& \exists x_1 Flight(Hamburg, x_1) \wedge Flight(x_1, x) \vee \\
& \exists x_1, x_2 Flight(Hamburg, x_2) \wedge Flight(x_2, x_1) \wedge Flight(x_1, x) \vee \\
\ldots &
\end{aligned}
$$

Example: Logic In DB Research

- **Finite Model Theory (FMT)** gives a proof for the impossibility to use FOL for recursive queries
- FMT models DBs as finite relational FOL structures

Example

- Flight table becomes structure $\mathfrak{A} = (D, \mathit{Flight}^{\mathfrak{A}}, \mathit{Hamburg}^{\mathfrak{A}}, \mathit{Berlin}^{\mathfrak{A}}, \dots)$
- Domain D: all constants in DB
- Constants named by themselves, e.g., $\mathit{Hamburg}^{\mathfrak{A}} = \mathit{Hamburg}$
- $\mathit{Flight}^{\mathfrak{A}} = \{(\mathit{Hamburg}, \mathit{Berlin}), (\mathit{Hamburg}, \mathit{NewYork}), \dots\}$

Example: Logic In DB Research

- Investigate all relevant reasoning problems w.r.t. finite models
  - Many properties for classical FOL do not hold
  - Also w.r.t. complexity
    $\Longrightarrow$ Calls for new techniques
- In particular: Investigate properties that all FOL queries have.

Theorem

*All FOL formulas are **local**.*
(Holds even for FOL extended with aggregation)

- Recursive queries are not local!

Example: Data Exchange (Lectures 5-6)

- Deals in a specific way with the integration of DBs

- Scenario
  - You have two DBs (source and target) on the same domain but different schemata S and T
  - You have some relationship specifications $\Sigma(T, S)$ for T and S
  - Aim: Answer queries over T to get answers with DBs over S
  - Subaims: Find (good) instances for T corresponding to given instances over S and answer over found solution set.

- And here comes logic
  - Language for specifying $\Sigma_{ST} \Longrightarrow$ Specific FOL formulas called tuple generating dependencies (tgds)
  - Criteria for goodness of solutions $\Longrightarrow$ universal model notion
  - How specify answers? $\Longrightarrow$ Certain answer semantics

Example: Data Exchange (Lectures 5-6)

- Deals in a specific way with the integration of DBs

- Scenario
  - You have two DBs (source and target) on the same domain but different schemata S and T
  - You have some relationship specifications $\Sigma(T, S)$ for T and S
  - Aim: Answer queries over T to get answers with DBs over S
  - Subaims: Find (good) instances for T corresponding to given instances over S and answer over found solution set.

- And here comes logic
  - Language for specifying $\Sigma_{ST} \Longrightarrow$ Specific FOL formulas called tuple generating dependencies (tgds)
  - Criteria for goodness of solutions $\Longrightarrow$ universal model notion
  - How specify answers? $\Longrightarrow$ Certain answer semantics

Example: Data Exchange

Example

- S: student(name)
- T: univ(sname, uname)
- Σ_{ST} : $student(x) \rightarrow \exists y\ univ(x, y)$

  "If something is a student in a S-DB, then there is an associated university in the T-DB"

- Example T-query: $Q(x) = \exists y.univ(x, y)$
- What should be the answers for given S-DB $I = \{\text{student(Frege)}\}$?
  cert(Q(x),I) = {Frege}

Example: Querying via Ontologies (Lectures 7-8)

- Ontologies as formal means to represent and reason over data
- Ontologies specify constraints and completeness rules
- Ontologies may have many models (open world assumption)
- May be used for access of heterogeneous data sources

- Appropriate ontology languages: Description Logics (OWL and variants)
  - Constants, concepts (unary predicates), roles (binary predicates)
  - Terminological axioms, e.g., $\mathit{Students} \sqsubseteq \mathit{Humans}$
  - Assertions axioms, e.g., $\mathit{Student}(\mathit{Frege})$
  - Description logics are **feasible fragments** of FOL

Example

- No university known for *Goedel*
- Completeness:
  $Student \sqsubseteq \exists hasUniv.University$
- Functionality constraint:
  $(func\ hasUniv)$

Table university

Student	Univ
Frege	U-Jena
Russell	U-London
Goedel	NULL
...	...

Example: Ontology Integration (Lectures 9-10)

- There exist many ontologies out there
- For some applications need to integrate ontologies
- Problem: Joining ontologies may lead to incoherences/inconsistencies

Example

Ontology A

- $Article \equiv \exists publ.Journal$
- $Journal \sqsubseteq \neg Proceedings$
- $(func\ publ)$

Ontology B

- $Article \equiv \exists publ.Journal \sqcup Proceedings$
- $publish(ab, procXY)$

- $O_A \cup O_B$ is inconsistent
- How to repair this?

Belief Revision

- Belief Revision deals with operators for revising theories under possible inconsistencies
- Investigates concrete revision operators
- Principles that these must fulfill (minimality etc.)
- Representation theorems

- Recent research how to adapt these for non-classical logics/ontologies

Streams (Lectures 11-12)

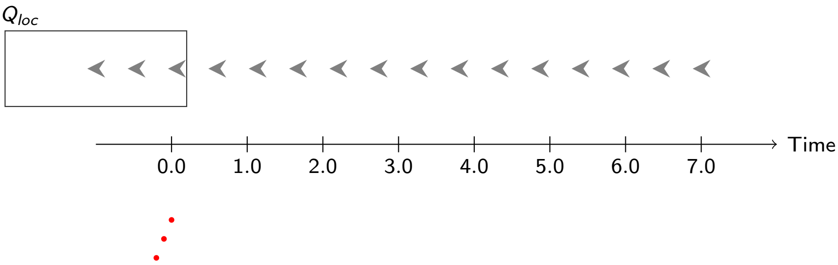
- "It's a streaming world" **(Ubiquity)**
  - Many data are temporal (sensor, event data)
  - Big data is mostly temporal data
- "Streams are forever" **(Potential Infinity)**
  - Streams are potentially infinite
  - One has to tame the infinite
  - Streams call for continuous **querying** (monitoring)
- "Order Matters" **(Sequentiality)**
  - Stream elements have an arriving order next to temporal order
  - Re-ordering or special sequencing may be needed

Lit: E. Della Valle. et al. It's a streaming world! Reasoning upon rapidly changing information. Intelligent Systems, IEEE, 24(6):83–89, nov.-dec. 2009.

Lit: J. Endrullis, D. Hendriks, and J. W. Klop. Streams are forever. Bulletin of the EATCS, 109:70–106, 2013.

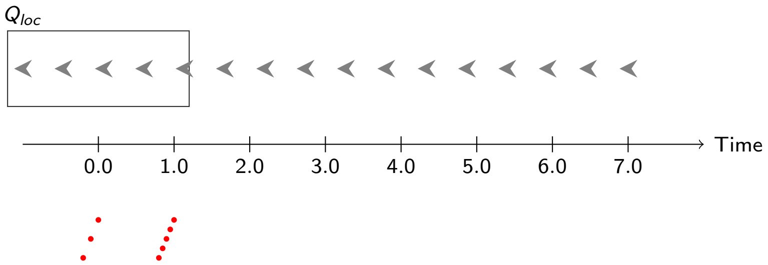
Lit: E. D. Valle et al. Order matters! Harnessing a world of orderings for reasoning over massive data. Semantic Web, 4(2):219–231, 2013.

Query With Sliding Window



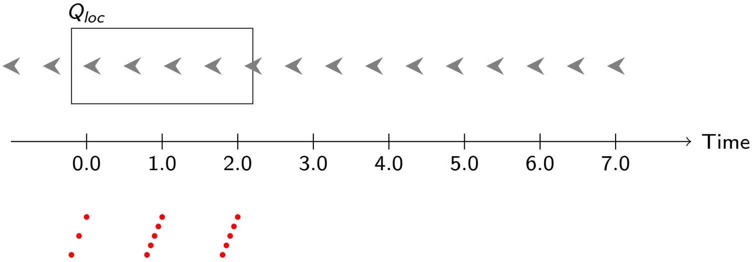
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



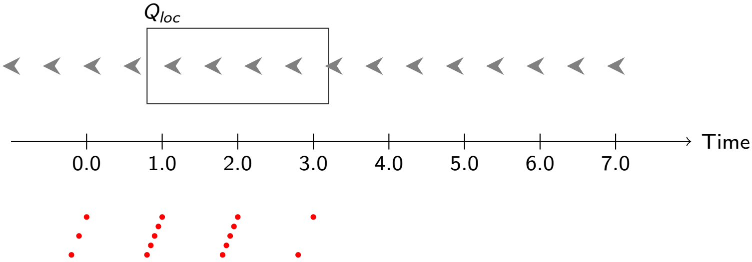
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



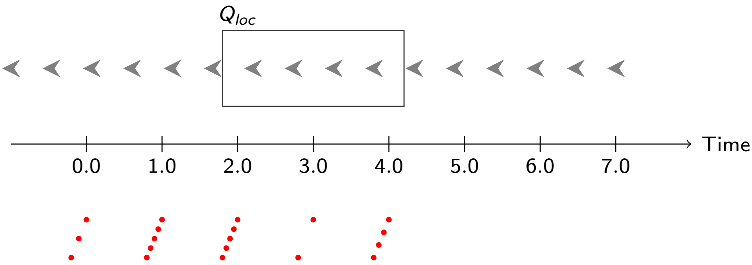
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



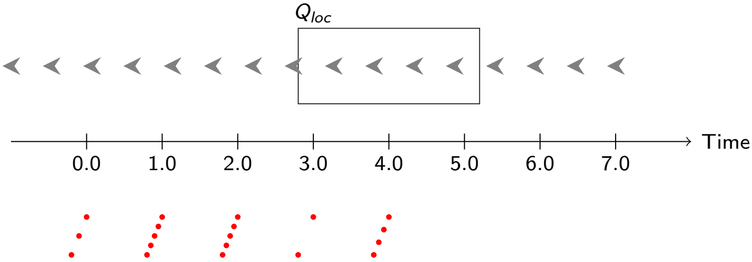
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



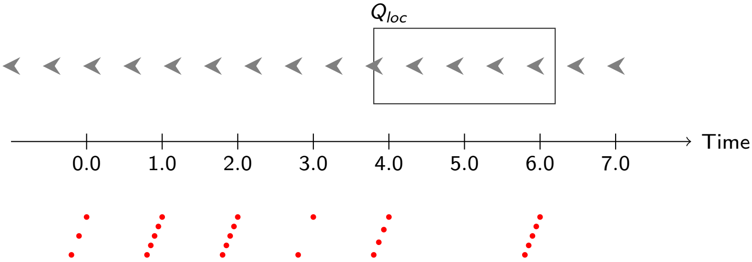
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



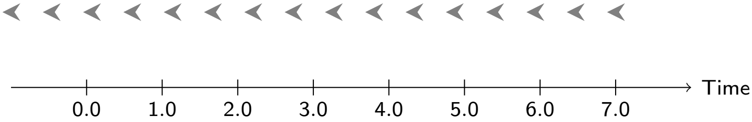
- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Query With Sliding Window



- Sliding window for taming the infinite
- **Query** window contents with local query Q_{loc}
- Example: Q_{loc} = Show all failure events in the window
- For High-Level Stream Processing: Incorporate background knowledge

Process Verification (Lectures 12-13)

- Verification of system behavior very important for industrial applications
- Model Checking mature theory with well-proven software implementations used in industry
  - given a system description (model) and (desired) specifications (axioms in (**temporal logics**))
  - Check whether specification is fulfilled by (all runs of) model

Example (Linear Temporal Logic)

- Excluding unwanted conditions for every time point

$$\Box \neg(\mathit{turbineTemp} > 90^\circ)$$

- Ensuring wanted conditions

$$\Box(\mathit{startTurbine} \rightarrow \diamond \mathit{TurbineIsRunning})$$

Process Verification

- Lift verification ideas/tools to verify business processes
- Challenges
  - Have to incorporate (large amounts of) data
    $\Longrightarrow$ artifact-centric approach (early 2000)
  - Finite state models not sufficient
    $\Longrightarrow$ finite state transducers

Exercise 1 (6 points)

Describe an example application or a computer science sub-area from your CS studies or from your job which exemplifies the "use" of some form of logic. In particular answer the following questions (on 2-3 slides in pdf):

1. What kind of logic is used?
2. What is its relation to FOL?
3. How/why is it used in the area/application?

- Send your solutions in one pdf file as presentation until Monday night, 24th of October 2016 to oezcep@ifis.uni-luebeck.de.
- You may work in pairs
- State your name, your study course (Studiengang) and your identity number (Matrikelnummer) at the title page