Semantic Web

Logical foundations

Acknowledgements to Pascal Hitzler, York Sure

„Logic is the Calculus of Computer Science“

~

„The central role of logic in computer science is comparable to the role of differential equations in the natural sciences.“

Applications

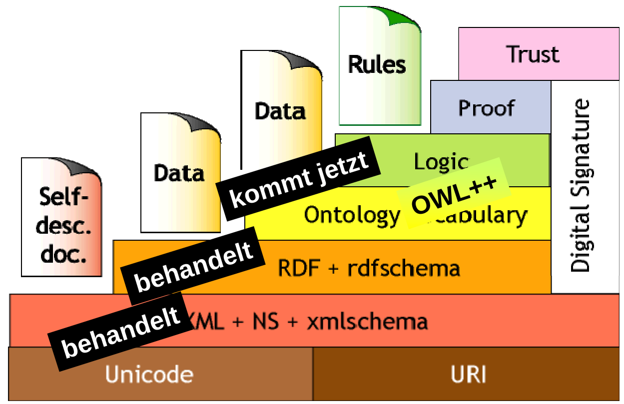
Calculus

Physics
Engineering sciences
Chemistry
Biology
Via statistics also in
  social sciences
  medicine
Etc.

Logik

Knowledge representation
Automated proofs
Cognitive robotics
Program verification
Semantics of programming languages
Databases
Data integration
Electronics
Etc.

The Semantic Web Layer Cake



Objectives of following lectures

1. Logics
2. Web Ontology Language OWL
3. OWL and rule languages

- Repetition of foundations
- Knowledge about established ontology languages and their backgrounds
- Basic understanding of automated reasoning
- Foundations of current research discussion (bachelor/master theses)

Inhalte der nächsten Vorlesungen

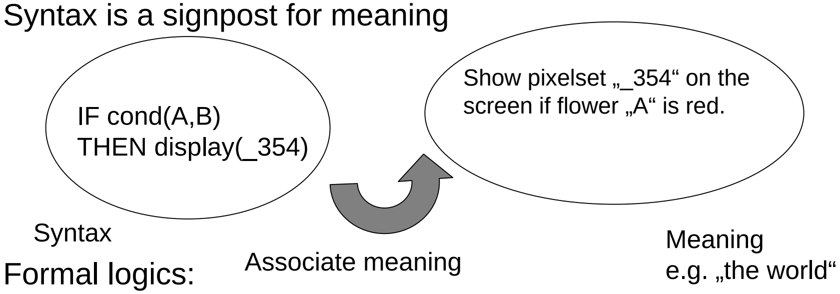
1. Logics
   - Propositional logics + first order predicate logics (FOL)
   - Syntax and semantics
2. Web Ontology Language OWL
   - OWL as description logics / FOL-fragment
   - Properties
3. Ontology Engineering
4. Automated reasoning FOL/OWL (somewhat later)

Logics

1. **What is semantics – generally speaking**
2. Syntax propositional logics + FOL
3. Model theoretic semantics
4. Properties of logics

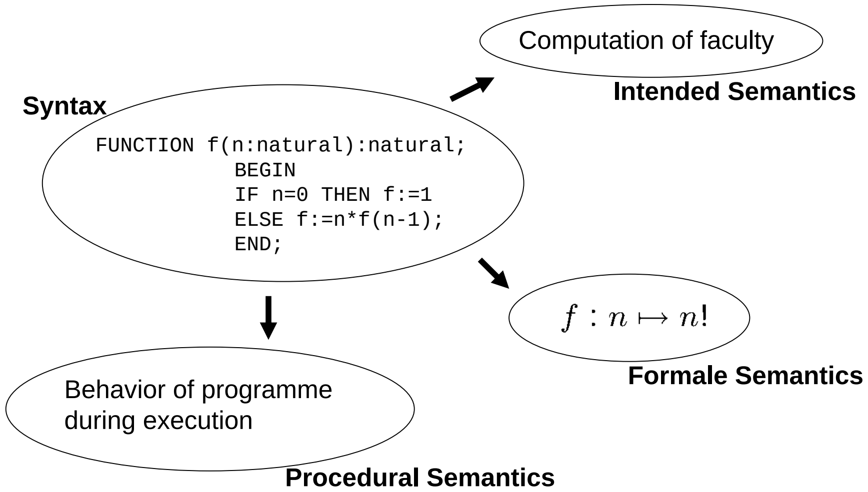
Syntax and Semantics

Syntax: Set of allowed sequences of characters/words
Semantik: Meaning associated with syntax
Syntax is a signpost for meaning

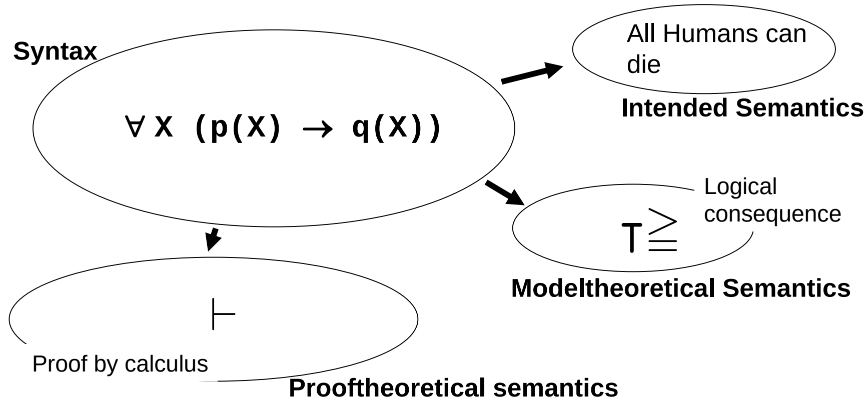


Formal logics:
Semantics of a statement is derived from its syntactic structures.
Frege: Meaning(„the apple is red“)=Meaning(„the apple“)+Meaning(„is red“)

What is semantics? Example programming language



Semantics of logics/knowledge representation language



Abstract forms of semantics

Game theoretic
Argumentation based
Algebraic
Category theory
Geometric
Automata theory
Denotational
Fix point semantics

Logics

1. **What is semantics – generally speaking**
2. **Syntax propositional logics + FOL**
3. Model theoretic semantics
4. Properties of logics

Propositional logics: Syntax

Junctor	*Name*	*Intuitive Bedeutung*
¬	Negation	„not“
∧	Konjunction	„and“
∨	Disjunction	„or“
→	Implication	„if – then“
↔	Biimplication	„exactly if then“

Predicate symbols/Variables in propositional logics, e.g. p, q, r, s, …
„correct“ composition of formula – use parentheses if in doubt:

$((p \land \neg q) \rightarrow s) \leftrightarrow \neg p$

$(p \lor \neg q) \lor (q \rightarrow \neg p)$

Precedences (we use): ¬ precedes ∧,∨ precedes →, ↔
Don’t hesitate to use extra parentheses ☺

Propositional logics: example

Simple propositions	*Modelling*
It rains.	r
The street will be wet.	w
The sun is green.	g

Composed propositions	Modellierung
If it rains, the street will be wet.	$r \rightarrow w$
If it rains and the street will not get wet, then the sun is green.	$(r \land \neg w) \rightarrow g$

First order predicate logics (FOL): Syntax: language elements

Quantor	*Name*	*Intuitive Bedeutung*
∀	All quantor, universal quantor	„for all“
∃	Existential quantor	„it exists“, „there is a“

- Junktors like in propositional logics
- Variables, e.g. X,Y,Z,…
- Constant symbols, e.g.. a, b, c, …
- Function symbols, e.g.. f, g, h, … (with arity)
- Relations-/Predicate symbols, e.g. p, q, r, … (with arity)

$(\forall X)(\exists Y)\ ((p(X) \land \neg q(f(X),Y)) \rightarrow r(X))$

FOL: Syntax

„correct“ composition of *terms* from variables, constant- and function symbols:

f(X), g(a,f(Y)), s(a), .(H,T), x_location(Pixel)

„correct“ composition of *Atoms* from relation symbols, the arguments of which are terms:

p(f(X)), q(s(a),g(a,f(Y))), add(a,s(a),s(a))
greater_than(x_location(Pixel),128)

„correct“ composition of *formula* from atoms, junctors and quantors:

(∀Pixel)(greater_than(x_location(Pixel),128) → red(Pixel))

Use parentheses if in doubt!
Quantify all varibles (closed formula only)!

FOL Syntax: Example *Addition*

(∀X)(∀Y)(∀Z)
(add(a,X,X)
∧ (add(X,Y,Z) → add(s(X),Y,s(Z)))
)

Intended semantics:
- a … 0 (zero)
- s … successor function/addition of one
- add(X,Y,Z) … „Z is the sum of X and Y“

FOL Syntax: Example *Lists*

(∀H)(∀T)(list([]) ∧ (list(T) → list(.(H,T))))
Informally: [] … empty list
.(H,T) … H is head, T rest
Also write: .(H,T) as [H|T]

(∀H)(∀T)
(member(a,[a|T])
∧ (member(a,T) → member(a,[H|T]))
)
Intended semantics:
member(x,list) … “x is element of list”

FOL Syntax: Example *Relationships*

(∀ X) (parent(X) ↔ (human(X) ∧ (∃Y) parent_of(X,Y)))

(∀ X) (human(X) → (∃Y) parent_of(Y,X))

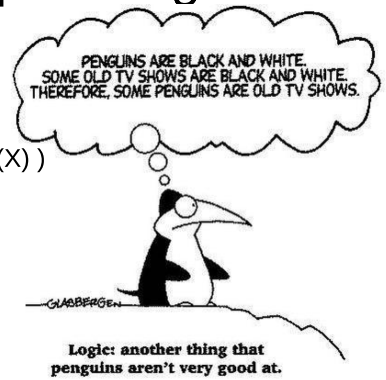
(∀ X) (orphan(X) ↔ (human(X) ∧
¬(∃Y) (parent_of(Y,X) ∧ alive(Y)))

(∀ X)(∀ Y)(∀ Z)
(uncle_of(X,Z) ↔ (brother_of(X,Y) ∧ parent_of(Y,Z)))

Intended semantics: as expected!

FOL Syntax: Example *Penguins*

((∀X)(penguin(X) → blackandwhite(X))
∧ (∃X)(oldTVshow(X) ∧ blackandwhite(X))
) → (∃X)(penguin(X) ∧ oldTVshow(X))



Intended semantics?

Logic can be used to show that penguins don’t think logically

Logics

1. What is semantics – generally speaking
2. Syntax propositional logics + FOL
3. **Model theoretic semantics**
4. Properties of logics

Propositional logics: model theoretic semantics

Interpretation:

Map each predicate symbol to {true,false}.

If F is a formula and I an interpretation then I(**F**) is a truth value, which is assigned from F and I using **truth tables**.

I(p)	I(q)	I(¬p)	I(p ∧ q)	I(p ∨ q)	I(p → q)	I(p ↔ q)
t	t	f	t	t	t	t
t	f	f	f	t	f	f
f	t	t	f	t	t	f
f	f	t	f	f	t	t

Propositional logics: model theoretical semantics

We write I |= F, if I(F)=true, and call the interpretation I a *model* of formula F.

Core notions:

valid (Tautology)

satisfiable (erfüllbar)

refutable

unsatisfiable/inconsistent/contradictory

Predicate Logics: Model theoretical semantics

Structure:

- Domain D (universe,…)
- Constant symbols are mapped onto elements of D
- Function symbols are mapped onto functions over D
- Relation symbols are mapped onto relations over D

This implies:

- Terms are interpreted as elements of D
- Relation symbols with their arguments are interpreted as being true or false
- Junctors/Quantors are treated to conform to truth tables

Predicate Logics: Model theoretical semantics Example

(∀X)(∀Y)(∀Z)

(add(a,X,X)

∧ (add(X,Y,Z) → add(s(X),Y,s(Z)))

)

Model I:

Domain: natural numbers N

I(a) = 0

I(s): n ↦ n+1

I(add(k,m,n))=true if and only if k+m=n.

I is a model of the formula.

Predicate Logics: Model theoretical semantics Example II

F = ((∀X)(penguin(X) → blackandwhite(X))

∧ (∃X)(oldTVshow(X) ∧ blackandwhite(X))

) → (∃X)(penguin(X) ∧ oldTVshow(X))

Interpretation I:

Domain:

a set M, which contains elements a,b,c.

… no constant or function symbols …

We show: The formula is refutable (i.e. it is not valid):

Assign: I(penguin)(a), I(blackandwhite)(a), I(oldTVshow)(b), I(blackandwhite)(b) true, I(oldTVshow)(a) false; then the formula is false for interpretation I.

We can use logics to show that penguins don't argue logically

Predicate Logics: Model theoretical semantics Example II

We write I |= F, if I(F)=true, and call the interpretation I a *model* of formula F.

Core notions:

- validity (Tautology)
- satisfiability
- refutability
- contradictory/unsatisfyable

Logical consequence/logical entailment

A *theory* T is a set of formula.

An interpretation I is a model for T, iff I |= G is true for all formulae G in T.

A formula F is a *logical consequence* from T, iff every model of T is also a model of F. We write T |= F.

Two formula F,G are *logically* (also *semantically*) *equivalent*, if {F} |= G and {G} $_T$ |= F. Then, we write F ≡ G.

Some logical equivalences

$F \wedge G \equiv G \wedge F$
$F \vee G \equiv G \vee F$
$F \rightarrow G \equiv \neg F \vee G$
$F \leftrightarrow G \equiv (F \rightarrow G) \wedge (G \rightarrow F)$

$\neg(F \wedge G) \equiv \neg F \vee \neg G$
$\neg(F \vee G) \equiv \neg F \wedge \neg G$

DeMorgan rules

$\neg\neg F \equiv F$

$F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$
$F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$

$\neg(\forall X)\, F \equiv (\exists X)\, \neg F$
$\neg(\exists X)\, F \equiv (\forall X)\, \neg F$

$(\forall X)(\forall Y)\, F \equiv (\forall Y)(\forall X)\, F$
$(\exists X)(\exists Y)\, F \equiv (\exists Y)(\exists X)\, F$

$(\forall X)\, (F \wedge G) \equiv (\forall X)\, F \wedge (\forall X)\, G$
$(\exists X)\, (F \vee G) \equiv (\exists X)\, F \vee (\exists X)\, G$

Logics

1. What is semantics – generally speaking
2. Syntax propositional logics + FOL
3. Model theoretic semantics
4. **Properties of logics**

Properties of predicate logics

- Monotony
  When the set of facts is enlarged, nothing what was previously concluded becomes invalid.
- Compactness
  For each consequence of a theory a finite subset of the theory is sufficient to draw it.
- Semi-decidability
  - All *true* consequences may be found if one searches long enough.
  - All contradictions may be found if one searches long enough (just negate all true consequences).
  - But: it is not possible to enumerate all sentences that are neither true consequences nor contradictions to the theory.
  - Hence, it is not possible to enumerate all sentences that are false consequences.

Properties of propositional logics

Include all properties of predicate logics; additionally:

- Decidability
  All *true consequences* may be found and all *false consequences* may be refuted if one searches long enough.
  I.e. there are theorem provers for propositional logics that always terminate.

Important fragments of first-order predicate logics

- Propositional Logics
- Datalog (Like pure Prolog, without function symbols)
  decidable
- Disjunctive Datalog (clauses without function symbols)
  decidable
- Definite programmes (pure Prolog)
  semi-decidable
- Description logics
  decidable (some of them)

  e.g. OWL → Coming next