Logik für Informatiker
Logic for computer scientists

What comes next?

Till Mossakowski

WiSe 2005/06

Beyond first-order logic

- many-sorted logic (variables, constants, predicates and functions have types)
  E.g.: $\forall n : Nat\ \forall l : List\ head(cons(n, l)) = n$

Beyond first-order logic

- many-sorted logic (variables, constants, predicates and functions have types)
  E.g.: $\forall n : Nat\ \forall l : List\ head(cons(n, l)) = n$
- partial function logic: $D(f(x))$ ("$f(x)$ is defined")

Beyond first-order logic

- many-sorted logic (variables, constants, predicates and functions have types)
  E.g.: $\forall n : Nat\ \forall l : List\ head(cons(n, l)) = n$
- partial function logic: $D(f(x))$ ("$f(x)$ is defined")
- higher-order logic: $\forall f : s \rightarrow t \ldots, \forall p : Pred(t) \ldots$

$$\forall u \forall v (Path(u, v) \leftrightarrow \forall R \ \{[\forall x \forall y \forall z (R(x, y) \wedge R(y, z) \rightarrow R(x, z)) \wedge \forall x \forall y (DirectWay(x, y) \rightarrow R(x, y))] \rightarrow R(u, v)\})$$

Modal and temporal logics

- modal logic:
  $\Box P$ (“necessarily P”) and $\Diamond P$ (“possibly P”)
  Other readings of $\Box P$:
  It ought to be that P
  It is known that P
  It is provable that P
  Always P (temporal logic)

Modal and temporal logics

- modal logic:
  $\Box P$ ("necessarily P") and $\Diamond P$ ("possibly P")
  Other readings of $\Box P$:
  It ought to be that P
  It is known that P
  It is provable that P
  Always P (temporal logic)
- temporal logic: $\Box P$ ("always in the future, P"), $\Diamond P$ ("sometimes in the future, P"), and $\bigcirc P$ ("in the next

step, P")

e.g. $\Box bank\_account > 0$ (very unrealistic)

Further modal and temporal logics

- temporal logic of actions (TLA): $\Box[state' = f(state)]_{state}$
  read: always in the future, either the state does not change, or the next state is f applied to the previous state

Further modal and temporal logics

- temporal logic of actions (TLA): $\Box[state' = f(state)]_{state}$
  read: always in the future, either the state does not change, or the next state is f applied to the previous state
- dynamic logic:
  $[p]P$ ("after every run of program p, P holds")
  $<p> P$ ("after some run of program p, P holds")

Further modal and temporal logics

- temporal logic of actions (TLA): $\Box[state' = f(state)]_{state}$
  read: always in the future, either the state does not change, or the next state is f applied to the previous state
- dynamic logic:
  $[p]P$ ("after every run of program p, P holds")
  $<p> P$ ("after some run of program p, P holds")
- spatial logics:

More exotic modal logics

- agent logics, e.g. ATL: agents A and B have the possibility to make a telephone call, if they cooperate

More exotic modal logics

- agent logics, e.g. ATL: agents A and B have the possibility to make a telephone call, if they cooperate
- logics for security, e.g. ABLP: A $\mathit{controls}$ P (“agent A has the permission to perform action P”)

Logics for knowledge representation/semantic web

- description logics, e.g. $\mathcal{ALC}$:
  $Elephant \doteq Mammal \sqcap \exists bodypart.Trunk \sqcap \forall color.Grey$
  abbreviates
  $$\forall x[Elephant(x) \leftrightarrow (Mammal(x) \wedge \exists y(bodypart(x,y) \wedge Trunk(y)) \wedge \forall z(color(x,z) \rightarrow Grey(z)))]$$

Multi-valued logics

- three-valued logics: truth values are true, false, and undefined

Multi-valued logics

- three-valued logics: truth values are true, false, and undefined
- object constraint logic (OCL)
  (for UML — the unified modeling language)

```
-- Managers get a higher salary than employees
inv Branch2:
  self.employee->forall(e | e <> self.manager
    implies self.manager.salary > e.salary)
```

Multi-valued logics (cont'd)

- fuzzy logic: truth values in the interval $[0, 1]$ correspond to different degrees of truth (e.g. Peter is quite tall, is tall, is very tall)

Even more exotic logics

- paraconsistent logics
  for databases, local inconsistency is o.k. and should not lead to global inconsistency

Even more exotic logics

- paraconsistent logics

  for databases, local inconsistency is o.k. and should not lead to global inconsistency

- non-monotonic logics

  new facts make previous arguments invalid, e.g.

  $Bird(x) \vdash CanFly(x)$

  $\{Bird(x), Penguin(x)\} \vdash \neg CanFly(x)$

Even more exotic logics

- paraconsistent logics

  for databases, local inconsistency is o.k. and should not lead to global inconsistency

- non-monotonic logics

  new facts make previous arguments invalid, e.g.

  $Bird(x) \vdash CanFly(x)$

  $\{Bird(x), Penguin(x)\} \vdash \neg CanFly(x)$

- linear logic (resource-bounded logic)

  $A \otimes A \vdash B$

  (we can prove B when we are allowed to use A twice)

Why do we need so many logics?

- different aspects of the complex world / of software systems
- one "big" logic covering everything would be too clumsy
- good news: most of the logics are based on propositional or first-order logics
- most of the logics have central notions in common

What is common to (most of) these logics?

What is common to (most of) these logics?

- A notion of language (or vocabulary of symbols, or signature)

What is common to (most of) these logics?

- A notion of language (or vocabulary of symbols, or signature)
- A syntax for sentences

What is common to (most of) these logics?

- A notion of language (or vocabulary of symbols, or signature)
- A syntax for sentences
- A notion of model

What is common to (most of) these logics?

- A notion of language (or vocabulary of symbols, or signature)
- A syntax for sentences
- A notion of model
- A notion of satisfaction, i.e. $M \models P$ (read: "M satisfies P", or "P holds in M")

What is common to (most of) these logics?

- A notion of language (or vocabulary of symbols, or signature)
- A syntax for sentences
- A notion of model
- A notion of satisfaction, i.e. $M \models P$ (read: "M satisfies P", or "P holds in M")
- A calculus $\mathcal{T} \vdash P$ (read "P is provable from $\mathcal{T}$)

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff

  for all models M with $M \models \mathcal{T}$, also $M \models P$

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff
  for all models M with $M \models \mathcal{T}$, also $M \models P$
- logical validity: $\models P$ iff for all models M, also $M \models P$

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff
  for all models M with $M \models \mathcal{T}$, also $M \models P$
- logical validity: $\models P$ iff for all models M, also $M \models P$
- satisfiability: $\mathcal{T}$ is satisfiable iff
  there is some M with $M \models \mathcal{T}$

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff
  for all models M with $M \models \mathcal{T}$, also $M \models P$
- logical validity: $\models P$ iff for all models M, also $M \models P$
- satisfiability: $\mathcal{T}$ is satisfiable iff
  there is some M with $M \models \mathcal{T}$
- formal consistency: $\mathcal{T}$ is formally consistent iff
  $\mathcal{T} \not\vdash P$ for some P

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff
  for all models M with $M \models \mathcal{T}$, also $M \models P$
- logical validity: $\models P$ iff for all models M, also $M \models P$
- satisfiability: $\mathcal{T}$ is satisfiable iff
  there is some M with $M \models \mathcal{T}$
- formal consistency: $\mathcal{T}$ is formally consistent iff
  $\mathcal{T} \not\vdash P$ for some P
- soundness of the calculus: $\mathcal{T} \vdash P$ implies $\mathcal{T} \models P$

What is common to all these logics? (cont'd)

- logical consequence: $\mathcal{T} \models P$ iff
  for all models M with $M \models \mathcal{T}$, also $M \models P$
- logical validity: $\models P$ iff for all models M, also $M \models P$
- satisfiability: $\mathcal{T}$ is satisfiable iff
  there is some M with $M \models \mathcal{T}$
- formal consistency: $\mathcal{T}$ is formally consistent iff
  $\mathcal{T} \not\vdash P$ for some P
- soundness of the calculus: $\mathcal{T} \vdash P$ implies $\mathcal{T} \models P$
- (sometimes) completeness: $\mathcal{T} \models P$ implies $\mathcal{T} \vdash P$

Multi logic systems

- The central notions common to all logics can be axiomatized
- Based on this meta-notion, multi-logic systems can be defined
- In Bremen, we also develop multi-logic tools

Next semester

Modal logic for computer scientists

MMISS evaluation of this course

Please (anonymously) fill out the questionaire and return it to us! (MZH 8070)

Abgabe der Übungsaufgaben

bis 28. Februar 2006