Chapter 9
Reasoning

Queries vs. Conclusions

- Query: (for which X) does something hold in a given database state?
- Conclusions: given some facts and some *knowledge*, does something hold?

Knowledge Bases

- general statements/logic formulas (cf. human knowledge and reasoning)
- often definitions: "adults are persons who are at least 18 years old", "parents are persons who have children", "every person is either male or female", "my uncles are the male siblings of my parents, and the husbands of the female siblings of my parents",
- can often, but not always, be represented as rules.
- Queries against knowledge bases are answered by *reasoning*, not by algebraic evaluation (although some reasoning can be implemented on an algebraic base).

9.1 Model Theory and Logical Entailment

For formalizing (and applying) reasoning, *logical entailment*,

$$\mathcal{F} \models \varphi$$

is needed:

- $\mathcal{F}$: a set of formulas, the specification
- φ: a formula
- does $\mathcal{F}$ *entail* φ,
  i.e., assumed that $\mathcal{F}$ holds, can we conclude that φ also holds?
- e.g. $\mathcal{F}$ the specification of the notion of "uncle", and a given database of persons:
  - does "Bob is an uncle of Alice" hold?
  - which persons (in the database) are uncles of Alice?
  - which persons (in the database!) are *not* uncles of Alice?
    - remark on closed world vs. open world

DATABASES VS. KNOWLEDGE BASES

- A *database (state)* is a *relational structure*.

  We can check whether a formula holds there, or for which values of X it holds (which is then a query).

  The *semantics* of a database is the *current database state*.

- A (first-order) *knowledge base* is a set of closed (first-order) formulas. It contains *facts*, but also other *formulas*.

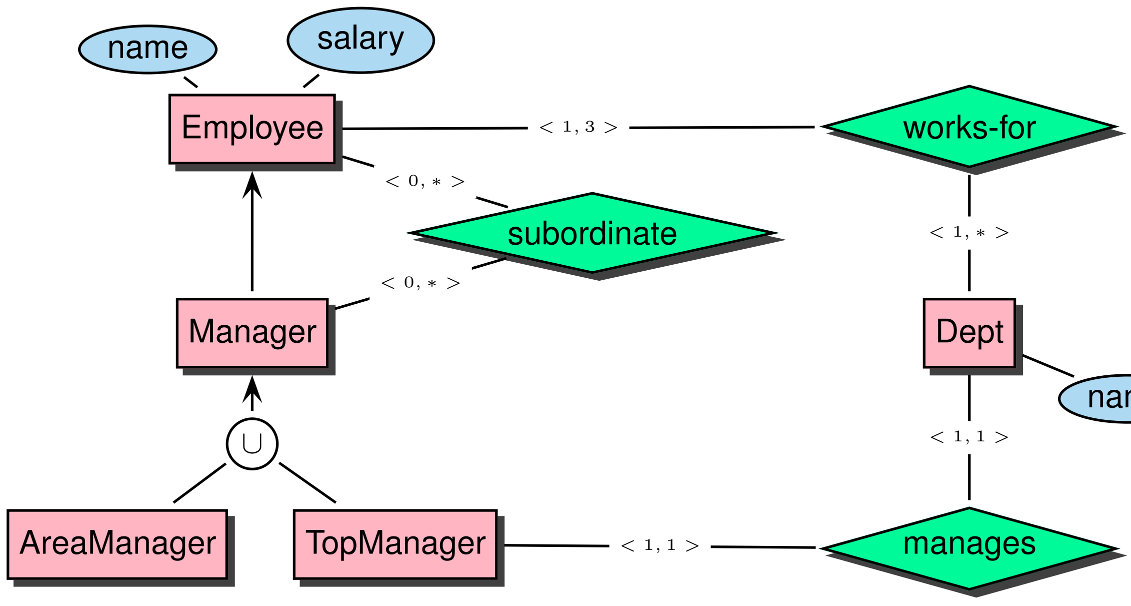
  We are interested if a knowledge base $\mathcal{K}$ *implies* a fact or a formula F. This means, if for *all* models $\mathcal{M}$ of $\mathcal{K}$, F must be true in $\mathcal{M}$:

  Whenever $\mathcal{M} \models \mathcal{K}$, does then $\mathcal{M} \models F$ hold?

  The semantics of a knowledge base (or in general a set of formulas) is the *set of all its models*.

- an intermediate form occurs when a database is extended by axiomatic formulas (subclasses etc.) or rules that can be used to derive additional facts.
  Then, the semantics is given by the model(s) of the database state and the rules.

EXAMPLE: ER DIAGRAM AS AN ONTOLOGY



(TopManager: leads a department;
AreaManager: intermediate group leaders in a department)

What can be “models” of this ontology? How do you represent them? Give an example.

Semantics: Set theory

- a class is a set of instances,
  Employee={alice, bob, john, mary, tom, larry} and Manager={alice, bob, john, mary},
  AreaManager={mary} and TopManager={alice, bob, john},
  Dept={sales, production, management}

  Constraints from subclasses:
  Manager = *AreaManager* $\cup$ *TopManager*
  Manager $\subseteq$ *Employee*
  AreaManager $\subseteq$ *Manager* and
  TopManager $\subseteq$ *Manager* (both redundant)

- an attribute is a set of pairs of (i) an instance and (ii) an element of a literal domain (constraint!)
  name =
  {(alice, "Alice"), (bob, "Bob"), (john, "John"), (mary, "Mary"), (tom, "Tom"), (larry, "Larry")}
  salary =
  {(alice, 70000), (bob, 60000), (john, 100000), (mary, 40000), (tom, 25000), (larry, 20000)}
  analogously for department names.

Semantics: Set theory (Cont'd)

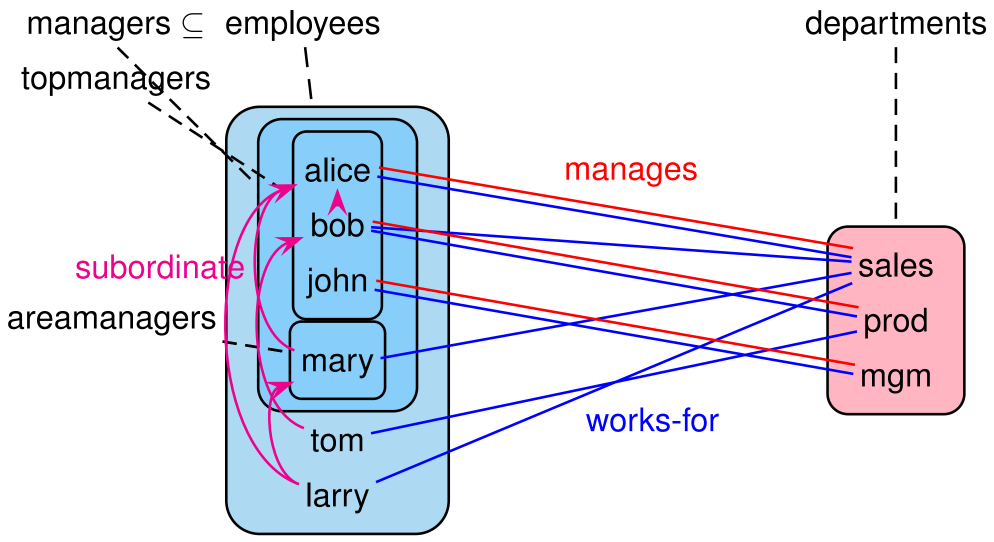
- a relationship is a pair of instances,
  (or a set of n-tuples, in case of n-ary relationships)
  works-for $\subseteq$ *Employee* $\times$ *Dept*

  works-for = *{(alice, sales), (mary, sales), (larry, sales), (bob, production),*
  (bob, sales), (tom, production), (john, management)}

  manages $\subseteq$ *TopManager* $\times$ *Dept*
  manages = *{(alice, sales), (bob, production), (john, management)}*
  subordinate $\subseteq$ *Employee* $\times$ *Manager*
  subordinate = *{(mary, alice), (bob, alice), (larry, mary), (larry, alice), (tom, bob)}*

- not so obvious: constraints coming from the cardinality specifications, e.g.,
  the set of top managers is a subset of the things that manage exactly one department,
  the set of employees is a subset of the things that work for at least one and at most three departments
  (see later after the discussion of first-order logic)

Semantics: Set theory – Graphical Illustration



- does manages ⊆ works-for hold in general?
- subordinate ⊇ works-for ∘ manages^{-1} ?
- subordinate ⊇ (works-for \ manages) ∘ manages^{-1} !! ("\" denotes set difference)
- the subordinates relationship wrt. area managers (larry→mary) cannot be derived but must be stated explicitly.

Adequateness of (extended) ER Diagrams

An ontology should give a concise characterization of a domain and its constraints.

- classes, key constraints
- subclasses (specialization/generalization, disjointness)
- ranges and domains of properties (e.g., that the domain of "manages" is not all employees but only the managers)
- cardinalities
- is manages ⊂ works-for ?
- is manages ∩ works-for = ∅ ?
- subproperty constraints cannot be expressed
- further constraints (e.g., employees that work for a department are subordinate to the department's manager) cannot be expressed.

Exercise

- Discuss alternatives for the cardinalities for "subordinate".

Alternative Semantics: Relational Model

Exercise: give a relational schema and the corresponding database state to the above ER diagram.

Relational Schema as an Ontology

- basically non-graphical, can be supported e.g. by dependency diagrams (cf. the Mondial documentation)
- no distinction between “classes” and “relationships”
- key constraints, foreign key/referential constraints
- keys/foreign key allow to guess classes vs. relationships
- cardinality “1”: functional dependencies (adding n:1-relationships into a table like department/manages country/capital)
- no general cardinality constraints
- sometimes: domain constraints (by foreign keys)
- no further (inter- and intra-relation) constraints

Example: Axiomatization of the “Company” Ontology

Consider again the ER diagram from Slide 503.

- give the *first-order signature* Σ of the ontology,
- formalize the constraints given in the
  - subclass constraints
  - range and domain constraints
  - cardinality constraints

  and
  - additional constraints/definitions that cannot be expressed by the ER model.

  (this set of formulas is called a first-order “theory” or “axiomatization” of the ontology)
- express the instance level as an interpretation of the signature Σ.

Example: Signature

- Classes are represented by unary predicates: Emp/1, Mgr/1, AMgr/1, TMgr/1, Dept/1.
- Attributes are represented by binary predicates: name/2, salary/2 (optionally, this could be modeled by unary functions)
- (binary) relationships are represented by binary relationships: wf/2, mg/2, sub/2.

Thus,
$\Sigma_{company} =$ {Emp/1, Mgr/1, AMgr/1, TMgr/1, Dept/1, name/2, salary/2, wf/2, mg/2, sub/2}.

Example: Subclass Constraints

$$\begin{aligned}
&\forall x : \mathsf{Mgr}(x) \rightarrow \mathsf{Emp}(x) \ , \\
&\forall x : \mathsf{AMgr}(x) \rightarrow \mathsf{Mgr}(x) \ , \\
&\forall x : \mathsf{TMgr}(x) \rightarrow \mathsf{Mgr}(x) \ , \\
&\forall x : \mathsf{Mgr}(x) \rightarrow (\mathsf{AMgr}(x) \vee \mathsf{TMgr}(x)) \ \text{ since declared as covering}
\end{aligned}$$

Example: Domain and Range Constraints

$$\begin{aligned}
&\forall x : (\exists n : \mathsf{name}(x, n) \rightarrow (\mathsf{Emp}(x) \vee \mathsf{Dept}(x))) \ , \\
&\forall x : (\exists s : \mathsf{salary}(x, s) \rightarrow (\mathsf{Emp}(x))) \ , \\
&\forall x, y : (\mathsf{sub}(x, y) \rightarrow (\mathsf{Emp}(x) \wedge \mathsf{Mgr}(y))) \ , \\
&\forall x, d : (\mathsf{wf}(x, d) \rightarrow (\mathsf{Emp}(x) \wedge \mathsf{Dept}(d))) \ , \\
&\forall x, d : (\mathsf{mg}(y, d) \rightarrow (\mathsf{Mgr}(y) \wedge \mathsf{Dept}(d))) \ .
\end{aligned}$$

Example: Cardinality Constraints

$$\begin{aligned}
&\forall m : (\mathsf{TMgr}(m) \rightarrow \exists d : \mathsf{mg}(m, d)) \ , \\
&\forall m, d_1, d_2 : ((\mathsf{mg}(m, d_1) \wedge \mathsf{mg}(m, d_2)) \rightarrow d_1 = d_2) \ , \\
&\forall d : (\mathsf{Dept}(d) \rightarrow \exists m : \mathsf{mg}(m, d)) \ , \\
&\forall d, m_1, m_2 : ((\mathsf{mg}(m_1, d) \wedge \mathsf{mg}(m_2, d)) \rightarrow m_1 = m_2) \ , \\
&\forall d : (\mathsf{Dept}(d) \rightarrow \exists x : \mathsf{wf}(x, d)) \ , \\
&\forall x : (\mathsf{Emp}(x) \rightarrow \exists d : \mathsf{wf}(x, d)) \ , \\
&\forall x : ((\exists d_1, d_2, d_3, d_4 : \mathsf{wf}(x, d_1) \wedge \mathsf{wf}(x, d_2) \wedge \mathsf{wf}(x, d_3) \wedge \mathsf{wf}(x, d_4)) \rightarrow \\
&\qquad (d_1 = d_2 \vee d_1 = d_3 \vee d_1 = d_4 \vee d_2 = d_3 \vee d_2 = d_4 \vee d_3 = d_4))
\end{aligned}$$

Example: Further Constraints

- a person is subordinate to the manager of each department he/she works for:
$$\forall x, y, d : \text{wf}(x, d) \wedge \text{mg}(y, d) \wedge x \neq y \rightarrow \text{sub}(x, y)$$
- should we have mg $\subseteq$ wf, or mg $\cap$ wf $= \emptyset$?
The first is OK: $\forall y, d : \text{mg}(y, d) \rightarrow \text{wf}(y, d)$
- add axioms that guarantee irreflexivity and transitivity for "subordinate":
$\forall x : \neg\text{sub}(x, x)$ and $\forall x, y, z : (\text{sub}(x, y) \wedge \text{sub}(y, z) \rightarrow \text{sub}(x, z))$.

Constraints that are not added

- cardinality of "subordinate"? "Every employee has a boss"
$$\forall x : \text{Emp}(x) \rightarrow \exists y : \text{sub}(x, y)$$
This causes a semantical problem with the boss: an infinite chain is needed - leading either to only infinite models, or a cycle.
- add an axiom that guarantees that the company has at least one employee:
$\exists x : \text{Emp}(x)$ – then the set of axioms is unsatisfiable.
- such investigations help to validate an ontology.
Ontology design tools allow to check for inconsistency, empty classes etc.

Axiomatization of the "company" scenario

Denote the conjunction of the above formulas by $\text{Axioms}_{Company}$.

- For any database/knowledge base $\mathcal{S}$ using this scenario, $\mathcal{S} \models \text{Axioms}_{Company}$ is required.
- a database then describes the individuals and their individual properties in this world.

Example: Instances

- The signature is extended by constant symbols for all *named* elements of the domain:

  $\Sigma_{my_company} := \Sigma_{company} \cup \{$Alice/f0, Bob/f0, John/f0, Mary/f0, Tom/f0, $\ldots$, Sales/f0, $\ldots\}$

  (Note: the signature symbols are capitalized, wereas alice, bob etc denote the elements of the domain).
- first-order structure $\mathcal{S} = (I, \mathcal{D})$ as ...
- Domain $\mathcal{D} = \{$alice, bob, john, mary, tom, larry, sales, prod, mgm$\}$
- map constant symbols (nullary function symbols) to $\mathcal{D}$:
  $I(\text{Alice}) = \text{alice},\ I(\text{Bob}) = \text{bob},\ \ldots,\ I(\text{Sales}) = \text{sales}, \ldots$.
- map unary predicates to subsets of the domain $\mathcal{D}$:
  $I(\text{Emp}) = \{$alice, bob, john, mary, tom, larry$\}$, $I(\text{Mgr}) = \ldots,\ \ I(\text{Dept}) = \ldots\ ,\ \ \ldots$,
- map binary predicates to subsets of $\mathcal{D} \times \mathcal{D}$:

  $I(\text{wf}) =$ {(alice, sales), (mary, sales), (larry, sales), (bob, prod), (bob, sales),
  (tom, prod), (john, mgm)}

  $I(\text{mg})$, $I(\text{sub})$, $I(\text{name})$, $I(\text{salary})$ see Slide 505.

Axiomatization of a Scenario

The axiomatization of a scenario (general formulas + a given instance) consists of

- the general axiomatization of the entity types/classes and relationships,
- literals describing the individuals (class membership and relationships),
- closure formulas that state that no other things/relationships exist.

DB Queries vs. Reasoning

- queries:
  - for which X, Y does $\mathcal{S} \models F(X, Y)$ hold?
  - against a single instance,
  - this allows algebraic evaluation (if the query can expressed in the algebra).
- reasoning:
  - for which X, Y does a specification $\mathcal{F}$ logically imply that $F(X, Y)$ holds?
  - There are all models of $\mathcal{F}$ considered (although, if it is a complete specification with closure axioms, there is only one model).
  - allows “querying” general specifications for e.g. guaranteeing correctness properties.

Example: Company Ontology

The axiomatization of “my company” with the given individuals is then

$$\begin{aligned}\text{Axioms}_{Company} \wedge\ & \text{Emp}(\text{Alice}) \wedge \text{Emp}(\text{Bob}) \wedge \ldots \wedge \text{Dept}(\text{Sales}) \wedge \ldots \wedge \\ & \text{wf}(\text{Alice}, \text{Sales}) \wedge \ldots \wedge \text{mg}(\text{Alice}, \text{Sales}) \wedge \ldots \wedge \\ & \forall x : \text{Emp}(x) \rightarrow (x = \text{Alice} \vee x = \text{Bob} \vee \ldots) \wedge \\ & \forall x : \text{Dept}(x) \rightarrow (x = \text{Sales} \vee x = \text{Prod} \vee \ldots) \wedge \\ & \forall x, y : \text{wf}(x, y) \rightarrow ((x = \text{Alice} \wedge y = \text{Sales}) \vee \ldots) \wedge \ldots\end{aligned}$$

Example: Instances (Alternative)

- alternatively, instead of constant symbols, all individuals can be described by *existentially* quantified variables:

$\text{Axioms}_{Company} \wedge$
$\exists alice, bob, \ldots, sales, \ldots : (\text{name}(alice, \text{“Alice”}) \wedge \ldots \wedge \text{Emp}(alice) \wedge \text{Emp}(bob) \wedge \ldots \wedge \text{Dept}(sales) \wedge$
$\ldots \wedge \text{Mgr}(alice) \wedge \ldots \wedge \text{wf}(alice, sales) \wedge \ldots \wedge \text{mg}(alice, sales) \wedge \ldots \wedge \text{sub}(mary, alice) \wedge \ldots)$

9.2 Logical Entailment

Definition 9.1
Let F and G two (closed) formulas over a signature Σ. We write

$$F \models G \quad (F \textit{ logically entails } G)$$

if for each structure $\mathcal{S}$ over Σ, if $\mathcal{S} \models F$ then also $\mathcal{S} \models G$. □

Example

$$\forall x : ((p(x) \rightarrow q(x)) \wedge (q(x) \rightarrow r(x))) \models \forall x : (p(x) \rightarrow r(x))$$

Logical Entailment as Proof

- usually F is a “large” conjunctive formula, containing the specification, and G is a “claim” to be shown to be a logical consequence of F.
- note: for an interpretation $\mathcal{I}$, the notation $\mathcal{I} \models F$ from the previous section can be regarded as representing $\mathcal{I}$ by its axiomatization $\phi_{\mathcal{I}}$ (cf. Slide 515) and then

$$\mathcal{I} \models F \quad \Leftrightarrow \quad \phi_{\mathcal{I}} \models F\ .$$

Logical Entailment in a Knowledge Base

- for a FOL knowledge base, it is not always necessary to give all facts explicitly,
- axioms and *some* “basic” facts are often sufficient,
- further facts can be proven/added to the KB by *logical entailment*,
- further universally quantified formulas can be derived,
- entailment is also relevant when verifying consistency (satisfiability) of an ontology specification.

(most of this: see later)

How to Prove Entailment?

- it is not necessary (and not possible) to compute all models of F to check if $F \models G$, instead
- prove it on the semantical level by *symbolic reasoning* ...

Logical Entailment: Example

Consider $\text{Axioms}_{Company} \wedge \text{mg}(\text{Alice}, \text{Sales})$.

Does Emp(Alice) hold in each model $\mathcal{S} = (\mathcal{D}, I)$ of $\text{Axioms}_{Company}$ (= is it logically entailed by the axioms)?

- $\mathcal{S} \models \text{mg}(\text{Alice}, \text{Sales})$ implies $(I(\text{Alice}), I(\text{Sales})) \in I(\text{mg})$, i.e., $(\text{alice}, \text{sales}) \in I(\text{mg})$.
- $\mathcal{S} \models \forall y, d : \text{mg}(y, d) \rightarrow \text{wf}(y, d)$ (axiom)
  implies that for all $d_1, d_2 \in \mathcal{D}$, $\mathcal{S} \models_{\{y/d_1, d/d_2\}} \text{mg}(y, d) \rightarrow \text{wf}(y, d)$ which means that if $\mathcal{S} \models_{\{y/d_1, d/d_2\}} \text{mg}(y, d)$, then also $\mathcal{S} \models_{\{y/d_1, d/d_2\}} \text{wf}(y, d)$. The former is equivalent to $(d_1, d_2) \in I(\text{mg})$ that we have shown above for $(\text{alice}, \text{sales})$. Thus, we know that $(\text{alice}, \text{sales}) \in I(\text{wf})$.
- With the same argument as above, use the axiom
  $\mathcal{S} \models \forall x, d : (\text{wf}(x, d) \rightarrow (\text{Emp}(x) \wedge \text{Dept}(d)))$ for concluding that $\text{alice} \in I(\text{Emp})$ which means that $\mathcal{S} \models \text{Emp}(\text{Alice})$.

Reasoning

- *prove* by symbolic reasoning that a formula is *implied* by a knowledge base: Algorithms for deriving entailed facts or formulas, or for checking entailment by automated, symbolic reasoning.

Validity and Decidability

- preferably use a decidable logic/formalism
- with a *complete* calculus/reasoning mechanism
- Propositional logic: decidable
- First-order logic: undecidable
- Horn subset (= positive rules, with a special model theory) of FOL: decidable; with negation in the body: still decidable
- 2-variable-subset of FOL: decidable
- Description Logic subsets of FOL: range from decidable to undecidable

A simple Natural Reasoning System

Inference rule “Modus Ponens”:

$$\frac{\forall \bar{x} : (P \rightarrow Q)\ ,\ P'}{\sigma(Q)} \text{ where } P' = \sigma(P) \text{ for a substitution } \sigma.$$

Consider again the derivation from Slide 519:

$$\frac{\forall y, d : (\text{mg}(y,d) \rightarrow \text{wf}(y,d))\ ,\ \text{mg}(\text{Alice}, \text{Sales})}{\text{wf}(\text{Alice}, \text{Sales})}$$

$$\frac{\forall x, d : (\text{wf}(x,d) \rightarrow (\text{Emp}(x) \wedge \text{Dept}(d)))\ ,\ \text{wf}(\text{Alice}, \text{Sales})}{\text{Emp}(\text{Alice}) \wedge \text{Dept}(\text{Sales})}$$

- forward-reasoning,
- uses only knowledge by implication rules “if ... then ...”.
  - no disjunction (disjunction is not part of daily common reasoning, but merely part of puzzles like Sudoku),
  - no existential quantification (reasoning with things that are known to exist but cannot be named explicitly)

... towards automated *symbolic reasoning*:

Prenex Form

Definition 9.2
A formula F is in prenex form if it has the form

$$F = Q_1 x_1 \; Q_2 x_2 \; \ldots \; Q_n x_n : G$$

where each $Q_i \in \{\forall, \exists\}$ is a quantifier and the x_i are variables, and G is quantifier-free. □

Theorem 9.1
For each formula F, there is an equivalent formula F' which is in prenex form. □

Proof: by induction. Pull quantifiers to the outside.

Next step: handling existentials:

- Consider $\forall x : (\text{Emp}(x) \to \exists d : \text{wf}(x,d))$ and $\forall d : (\text{Dept}(d) \to \exists m : \text{mg}(m,d))$
- for $x =$ Alice: "the department Alice works for", and "the person that manages this department"

$\Rightarrow$ implicit way to name things: $f_{dept}(\text{Alice})$ and $f_{manages}(f_{dept}(\text{Alice}))$.

SKOLEMIZATION

Consider formula F from Example 8.6(3): $F = \forall x \exists y : p(x,y)$.

When talking about models of it, "given a certain x_1, there is an y_1 such that ...".

One can imagine a (new) function f which returns "the (or one of them, if there are many) y for a given x": $f(x_1) := y_1$.

Definition 9.3 (Skolem Form)
[after Thoralf Skolem, a Norwegian Mathematician]

For a formula F in prenex form, its Skolem form $sk(F)$ is defined as follows:

For each subformula of the form $G = \exists y : H(x_1, \ldots, x_n, y)$ where the x_i are the free variables of H that are universally quantified by a subformula F' of F that contains G, replace each occurrence of y by the term $f(x_1, \ldots, x_n)$ where f is a new function symbol. □

Example 9.1
Consider $F = \forall x \exists y \forall z : p(x,y,z)$.
For skolemizing y consider $G(x) = \exists y \forall z : p(x,y,z)$ and replace y by $f(x)$.

$sk(\forall x \exists y \forall z : p(x,y,z)) = \forall x, z : p(x, f(x), z)$ □

Notes

- The definition is originally applied only to prenex normal form (i.e. all quantifiers on top, quantifier-free body);
- it holds in the same way for formulas that are not in prenex form (but the proof uses prenex form).
- There is an improvement:
  take only those universally quantified variables that are free in the body of the respective quantified subformula $\exists y : F(x_{i_1}, \ldots, x_{i_m})$.
- Further examples: Slide 528.

Usage of Skolemization

The formula $sk(F)$ is obviously not equivalent with F (it even uses an extended signature), but:

Theorem 9.2
For every formula F (in prenex form), $sk(F)$ is satisfiable if and only F is satisfiable. $\square$

Idea: extend the interpretation I with the new function symbols by mapping $f(d_1, \ldots, d_n)$ to a $d \in \mathcal{D}$ which exists for the given $x_1, \ldots, x_n$.

Proof ... needs a definition and a lemma before:

Definition 9.4 (Substitution)
A substitution is a mapping σ : Variables $\to$ Term$_\Sigma$.

For a formula F, variables $x_1, \ldots, x_n$ and terms $t_1, \ldots, t_n$, the application of $\sigma = [x_1/t_1, \ldots, x_n/t_n]$ to F, written as $F[x_1/t_1, \ldots, x_n/t_n]$, replaces every free occurrence of x_i in F by t_i.

A substitution is collision-free *if the mapped variables do not occur in any of the replacement terms.* $\square$

Example: $\sigma = [x/f(a), y/g(v, 3), z/f(g(a, w))]$ is collision-free, and $\sigma(p(x, y, z)) = p(f(a), g(v, 3), f(g(a, w)))$.

Lemma 9.1 (Substitution Lemma)
For every structure $\mathcal{S}$, and every variable assignment β, every terms s, t, every variable x, and every formula F,

- $\mathcal{S}(s[x/t], \beta) = \mathcal{I}(s, \beta_x^d)$ *where* $d = \mathcal{S}(t, \beta)$,
- $\mathcal{S} \models_\beta F[x/t] \Leftrightarrow \mathcal{S} \models_{\beta_x^d} F$ *where* $d = \mathcal{S}(t, \beta)$. $\square$

(Proof by structural induction over s and F)

Proof of Theorem 9.2

Induction over the number of replacements, top-down (replacing outer $\exists$-quantifiers first).
Consider F in prenex form
$F = \forall x_1, \ldots, x_n \exists y : G(x_1, \ldots, x_n, y)$, and its skolemization
$F' = \forall x_1, \ldots, x_n \exists y : G[y/f(x_1, \ldots, x_n)]$.

"$\Rightarrow$": Assume that F is satisfiable; there exists a structure $\mathcal{S} = (I, \mathcal{D})$ s.t. $\mathcal{S} \models F$.

Thus, for all $d_1, \ldots, d_n \in \mathcal{D}$,

$$\mathcal{S} \models_\beta \exists y : G(x_1, \ldots, x_n, y)$$

where $\beta = \{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$.
This is exactly the case if there is a $d \in \mathcal{D}$ (dependent on the d_i) s.t.

$$\mathcal{S} \models_{\beta_y^d} G(x_1, \ldots, x_n, y) \;.$$

Define a new structure $\mathcal{S}' = (I', \mathcal{D})$ such that I' coincides with I wherever I is defined, and defines the new n-ary function symbol f to map every n-tuple $(d_1, \ldots, d_n)$ to the above d (which depends on the d_i).
(using the Axiom of Choice which guarantees the existence of this function).

Proof of Theorem 9.2

"$\Rightarrow$" (cont'd): By the substitution lemma,

$$\mathcal{S}' \models_\beta G(x_1, \ldots, x_n, y)[y/f(x_1, \ldots, x_n)$$

an since this holds for all $d_1, \ldots, d_n \in \mathcal{D}$,

$$\mathcal{S}' \models \forall x_1, \ldots, \forall x_n : G(x_1, \ldots, x_n, y)[y/f(x_1, \ldots, x_n)],$$

i.e., $\mathcal{S}' \models F'$ – and F' is satisfiable (and there is a constructive description how (a possible) $\mathcal{S}'$ is obtained by extending $\mathcal{S}$).

"$\Leftarrow$": analogously.
$\mathcal{S}'(f(x_1, \ldots, x_n))$ is the "witness" element of the domain that satisfies the $\exists y$.

What is this good for??

- symbolic reasoning: the skolemized formula is sufficient to develop a "typical" model of the formula with the relationships between its constants in *tableau proofs*.

Example 9.2

Consider the formula

$$F = \forall x \exists y : (p(x,y) \wedge \exists z : (r(z) \wedge \neg q(x,y,z)))$$

- *prenex form:* $\forall x \exists y \exists z : (p(x,y) \wedge r(z) \wedge \neg q(x,y,z))$
  (aside: this is the same logical transformation as the "push-into-exists" rule of RANF)
- *skolemize with* f_y *and* f_z*:*
  $sk(F) = \forall x : (p(x, f_y(x)) \wedge r(f_z(x)) \wedge \neg q(x, f_y(x), f_z(x)))$.
- *understand that it is sufficient to describe* z *as* $f(x)$*, not as a function of* x *and* y*:*
  with $y \to f_1(x)$ *and* $z \to f_2(x,y)$*, follows* $z \to f_2(x, f_1(x))$ *which is effectively a function only of* x. □

Example 9.3

Consider the formula

$$F = (\forall x_1 \exists y : p(x_1, y)) \wedge (\forall x_2 \exists z : (q(x_2, z))$$

- *prenex form (one alternative):* $\forall x_1, x_2 \exists y, z : (p(x_1, y) \wedge q(x_2, z))$
- *skolemize with* f_y *and* f_z*:*
  $\forall x_1, x_2 : (p(x_1, f_y(x_1, x_2)) \wedge q(x_2, f_z(x_1, x_2)))$
  which is a lot longer than necessary!
- *skolemize without intermediate prenex step:*
  $(\forall x_1 : p(x_1, f_y(x_1))) \wedge (\forall x_2 : q(x_2, f_z(x_2)))$ □

Aside: Skolem Normal Form

Definition 9.5 (Skolem Normal Form)
A formula is in Skolem Normal Form if it is

- *closed (i.e., no free variables),*
- *of the form* $\forall x_1 \ldots \forall x_n : B$,
- *and* B *is in* conjunctive normal form (CNF)*, i.e., of the form*
  $(a_{11} \vee \ldots \vee a_{1k_1}) \wedge \ldots \wedge (a_{m1} \vee \ldots \vee a_{mk_m})$ □

- every formula F can be transformed in an equivalent formula F' in Skolem Normal Form.
- Skolem Normal Form is used for *Resolution Proofs*.
- in this lecture, we will not apply resolution to arbitrary inputs, but only to *logical rules* (Datalog rules) – which come automatically in CNF (and without function symbols).
- the idea of skolemization “on demand” is also used in Tableau proofs.

Reasoning

- *prove* by symbolic reasoning if a formula F is *implied* by a knowledge base $\mathcal{K}$ (which is a set of closed formulas):
  $$\mathcal{K} \models F \ \ ?$$
- Equivalently, let K denote the conjunction of all formulas in $\mathcal{K}$:
  $\emptyset \models K \rightarrow F$ does $K \rightarrow F$ hold in all interpretations?
- Equivalently,
  show that there is no interpretation where $\neg(K \rightarrow F)$ holds.
- Equivalently,
  show that $\neg(K \rightarrow F)$ (which is the same as $K \wedge \neg F$, i.e. a model of $K \wedge \neg F$ would be a model of K and not of F)   is unsatisfiable.

⇒ try to systematically develop a model of $K \wedge \neg F$.
If this fails, then, $K \wedge \neg F$ is unsatisfiable.

9.3 First Order Tableau Calculus

- Systematic construction of a model of a formula.
- Goal: show that this is not possible. Otherwise a counterexample is generated.
- counterexamples can be interpreted as answers to a query.

Start the tableau with a set $\mathcal{F}$ of formulas:

$$\boxed{\dfrac{\text{input set } \mathcal{F}}{F \quad \text{for all } F \in \mathcal{F}}}$$

The tableau is then extended by expansion rules.

Tableau Rules

Original Definition: *Raymond Smullyan: First-Order Logic. Springer, New York, 1968.*

α-rule (conjunctive):

$$\frac{F \wedge G}{\begin{array}{c}F\\G\end{array}} \qquad \frac{\neg(F \vee G)}{\begin{array}{c}\neg F\\\neg G\end{array}}$$

β-rule (disjunctive):

$$\frac{F \vee G}{F \mid G} \qquad \frac{\neg(F \wedge G)}{\neg F \mid \neg G}$$

γ-rule (universal):

$$\frac{\forall x : F}{F[X/x]} \qquad \frac{\neg\exists x : F}{\neg F[X/x]}$$

where X is a new variable.

Closure Rule:

$$\frac{\begin{array}{c}\sigma(A)\\\neg\sigma(A)\end{array}}{\perp}$$

(σ a substitution)

apply σ to the whole tableau.

δ-rule (existential):

$$\frac{\exists x : F}{F[f(\text{free}(F))/x]} \qquad \frac{\neg\forall x : F}{\neg F[f(\text{free}(F))/x]}$$

where f is a new *Skolem function symbol*.
Note: δ-Rule according to *Reiner Hähnle, Peter H. Schmitt: The Liberalized delta-Rule in Free Variable Semantic Tableaux. J. Autom. Reasoning 13(2): 211-221 (1994)*

Result

Definition 9.6
A branch T in a tableau $\mathcal{T}$ is closed*, if it contains the formula $\bot$.*
A tableau $\mathcal{T}$ is closed *if every branch is closed.* □

Correctness

Definition 9.7
A Tableau $\mathcal{T}$ is satisfiable *it there exists a structure $\mathcal{S} = (I, \mathcal{D})$ such that for every assignment β of the free variables there is a branch T in $\mathcal{T}$ such that $\mathcal{S} \models_\beta T$ holds.* □

Theorem 9.3
If a tableau $\mathcal{T}$ is satisfiable, and $\mathcal{T}'$ is obtained from $\mathcal{T}$ by application of one of the above rules, then $\mathcal{T}'$ is also satisfiable. □

Examples, Proof: to do in the lecture, sketch of two cases on Slide 535.

Issues: completeness of the method (only possible for decidable logics) and termination of the algorithm: how to detect when a tableau cannot be closed, and to restrict the expansion to promising rule applications.

Correctness of the FOL Tableau Calculus: Proof Sketch

Assume $\mathcal{T}$ satisfiable; $\mathcal{T}'$ obtained from applying a tableau rule. We show only two cases:

- Disjunction: Application of the rule to a formula of the form $A \vee B$. There is an interpretation $\mathcal{M}$ such that for each assignments β of free variables, there is some branch T (= the set of formulas on this branch) such that $\mathcal{M} \models_\beta T$. If T is not the branch of $\mathcal{T}$ that is extended in this step, T does not change. Otherwise, $M \models_\beta A \vee B$. By definition, $\mathcal{M} \models_\beta A$ or $\mathcal{M} \models_\beta B$. Thus, for (at least one) one of the two branches, T_1^* or T_2^* obtained from the application, $\mathcal{M} \models_\beta T^*$.
- Existential: Application of the rule to a formula of the form $\exists y : F(X_1, \ldots, X_n, y)$ to a branch T. Again, consider any β (which assigns $\beta(X_1), \ldots, \beta(X_n)$ to the free variables in F) such that $\mathcal{M} \models_\beta T$.

  This means, for every $\beta(X_1), \ldots, \beta(X_n)$, there is some element of the universe that “fits” for the existential formula. Extend the signature with a new n-ary “Skolem” function f_F that takes the values of $X_1, \ldots, X_n$ as input and is interpreted to return the appropriate element (and that returns an arbitrary value for those β' where $\mathcal{M} \not\models_{\beta'} T$).

  The extended branch T^* appends $F(X_1, \ldots, X_n, f_F(X_1, \ldots, X_n))$ to T.

  For the extended interpretation $\mathcal{M}'$ (which is the same as $\mathcal{M}$ except for the new function), $\mathcal{M}' \models_\beta T^*$ whenever $\mathcal{M} \models_\beta T$.

Tableau Calculus: Example

Consider again the derivation from Slide 519: Does

$\text{Axioms}_{Company} \wedge \text{mg}(\text{Alice}, \text{Sales})$. imply $\text{Emp}(\text{Alice})$?

- start the tableau with $\text{Axioms}_{Company}$, $\text{mg}(\text{Alice}, \text{Sales})$,   and the negated claim $\neg\text{Emp}(\text{Alice})$,
- see tableau next slide.
- this example: follow human reasoning:
  - the proof steps are known,
  - "apply" $\forall y, d : (\text{mg}(y, d) \rightarrow \text{wf}(y, d))$ with $y/\text{Alice}, d/\text{Sales}$, "obtain" $\text{wf}(\text{Alice}, \text{Sales})$
  - "apply" $\forall x, d : (\text{wf}(x, d) \rightarrow (\text{Emp}(x) \wedge \text{Dept}(d)))$ for $x/\text{Alice}, d/\text{Sales}$ and obtain $\text{Emp}(\text{Alice})$.
- the tableau illustrates the application of "rules":
  - close left branch "not body" immediately, propagate closure substitution to the right branch.

$\Rightarrow$ illustrative, but naive example driven by human *forward reasoning*.

Tableau Calculus: Example

$\forall y, d : (\text{mg}(y, d) \rightarrow \text{wf}(y, d))$

$\forall x, d : (\text{wf}(x, d) \rightarrow (\text{Emp}(x) \wedge \text{Dept}(d)))$

$\text{mg}(\text{Alice}, \text{Sales})$

$\neg\text{Emp}(\text{Alice})\ \ (**)$

|

$\text{mg}(Y_1, D_1) \rightarrow \text{wf}(Y_1, D_1)$

/ \

Left: $\neg\text{mg}(Y_1, D_1)$

$\Box\{Y_1 \rightarrow \text{Alice}, D_1 \rightarrow \text{Sales}\}$

Right: $\text{wf}(Y_1, D_1)$

$\text{wf}(\text{Alice}, \text{Sales})$

|

$\text{wf}(X_2, D_2) \rightarrow (\text{Emp}(X_2) \wedge \text{Dept}(D_2)))$

/ \

Left: $\neg\text{wf}(X_2, D_2)$

$\Box\{X_2 \rightarrow \text{Alice}, D_2 \rightarrow \text{Sales}\}$

Right: $\text{Emp}(X_2) \wedge \text{Dept}(D_2)$

$\text{Emp}(\text{Alice}) \wedge \text{Dept}(\text{Sales})$

$\text{Emp}(\text{Alice})\ \ (**)$

$\text{Dept}(\text{Sales})$

$\Box(**)$

Automated Reasoning

- do not close branches immediately by replacing variables
  - if there are multiple possible closing substitutions, keep the variable until the whole tableau can be closed,
- this also illustrates the use of *skolem functions* better.

EXAMPLE: TABLEAU EXPANSION FOR AN EXISTENTIAL VARIABLE

Consider again the Company scenario. Show: for every employee x, there is an employee y ($x = y$ allowed) such that sub(x, y) holds. (sketch: for every employee x there is a at least a "primary" department $f_{dept}(x)$ where this person works, and every department d has a manager $f_{mg}(d)$ that manages the department and that thus is a subordinate of x).

Note that in case that x works in several departments, any of them can be chosen for $f_{dept}(x)$. e is subordinate to $f_{mg}(f_{dept}(x))$.

Tableau: next slide.

- again: followed human reasoning steps.
- automated reasoning: how to choose which axioms to use?

```
                      $\forall x : (\text{Emp}(x) \rightarrow \exists d : \text{wf}(x, d))$
                      $\forall d : (\text{Dept}(d) \rightarrow \exists m : \text{mg}(m, d))$
                  $\forall x, d : (\text{wf}(x, d) \rightarrow (\text{Emp}(x) \wedge \text{Dept}(d)))$ ,
                  $\forall x, y, d : \text{wf}(x, d) \wedge \text{mg}(y, d) \rightarrow \text{sub}(x, y)$
                      $\exists e : \text{Emp}(e) \wedge \neg\exists y : \text{sub}(e, y)$   claim – refute it
                                  |
                      $\text{Emp}(e_0) \wedge \neg\exists y : \text{sub}(e_0, y)$  $e_0$ from Skolemization
                              $\text{Emp}(e_0)$
                           $\neg\exists y : \text{sub}(e_0, y)$
                            $\neg\text{sub}(e_0, Y_0)$
                                  |
                 $\text{Emp}(X_1) \rightarrow \exists d : \text{wf}(X_1, d))$
               /                                    \
       $\neg\text{Emp}(X_1)$                           $\exists d : \text{wf}(X_1, d))$
   close that only later ...               $\text{wf}(X_1, f_{dept}(X_1))$
                    $\text{wf}(X_2, D_2) \rightarrow (\text{Emp}(X_2) \wedge \text{Dept}(D_2)))$
                         /                              \
                  $\neg\text{wf}(X_2, D_2)$                  $\text{Emp}(X_2) \wedge \text{Dept}(D_2)$
        $\square$ $\{X_2 \rightarrow X_1, D_2 \rightarrow f_{dept}(X_1)$           $\text{Emp}(X_1)$
                                                   $\text{Dept}(f_{dept}(X_1))$
                                                          /
                      $\text{Dept}(D_3) \rightarrow \exists m : \text{mg}(m, D_3))$
                         /                        \
                  $\neg\text{Dept}(D_3)$                $\exists m : \text{mg}(m, D_3))$
            $\square$ $\{D_3 \rightarrow f_{dept}(X_1)\}$      $\text{mg}(f_{mgr}(D_3), D_3)$
                                          ... and now replace $D_3$
                                                   /
                      $\text{mg}(f_{mgr}(f_{dept}(X_1)), f_{dept}(X_1))$
                                        /
             $(\text{wf}(X_4, D_4) \wedge \text{mg}(Y_4, D_4)) \rightarrow \text{sub}(X_4, Y_4)$
                 /                                              \
       $\neg(\text{wf}(X_4, D_4) \wedge \text{mg}(Y_4, D_4))$                        $\text{sub}(X_4, Y_4)$
          /                          \                  $\text{sub}(X_1, f_{mgr}(f_{dept}(X_1)))$
   $\neg\text{wf}(X_4, D_4)$                $\neg\text{mg}(Y_4, D_4)$                         |
$\square\{X_4 \rightarrow X_1,$         $\square\{Y_4 \rightarrow f_{mgr}(f_{dept}(X_1)),$     close now everything by
   $D_4 \rightarrow f_{dept}(X_1)\}$       $D_4 \rightarrow f_{dept}(X_1)\}$            $\{X_1 \rightarrow e_0,$
                                                        $Y_0 \rightarrow f_{mgr}(f_{dept}(e_0))\}$
```

Tableau Calculus: Example

Consider again the tableau proof from Slide 536:

Does Axioms$_{Company}$ $\wedge$ mg(Alice, Sales) imply Emp(Alice)?

- again, start the tableau with Axioms$_{Company}$, mg(Alice, Sales), and the negated claim $\neg$Emp(Alice).
- automated reasoning: how to choose which axioms to use?
- consider the "goal" $\neg$Emp(Alice):
  clearly, some formula (here: rule) that derives Emp(Alice) is needed. Start with expanding $\forall x, d$: (wf$(x, d) \rightarrow$ (Emp$(x) \wedge$ Dept(d))).
- close the right branch with x/Alice, get a new "goal" $\neg$wf(Alice, D) in the left branch. Now, some formula (here: rule) that derives wf(Alice, _) is needed. Expand $\forall y, d$: (mg$(y, d) \rightarrow$ wf(y, d)).
- see tableau next slide.

Tableau Calculus: Example

Axioms$_{Company}$

mg(Alice, Sales)

$\neg$Emp(Alice) $(**)$

|

$\forall x, d$: (wf$(x, d) \rightarrow$ (Emp$(x) \wedge$ Dept(d)))

wf$(X_1, D_1) \rightarrow$ (Emp$(X_1) \wedge$ Dept(D_1))

/ \

$\neg$wf(X_1, D_1) Emp$(X_1) \wedge$ Dept(D_1)

| Emp(X_1) $(**)$

$\neg$wf(Alice, D_1) Dept(D_1)

$\forall y, d$: (mg$(y, d) \rightarrow$ wf(y, d)) $\square\{X_1/$Alice$\}(**)$

mg$(Y_2, D_2) \rightarrow$ wf(Y_2, D_2)

/ \

$\neg$mg(Y_2, D_2) wf(Y_2, D_2)

| $\square\{Y_2 \rightarrow$ Alice, $D_2 \rightarrow D_1\}$

$\neg$mg(Alice, D_1)

$\square\{D_1 \rightarrow$ Sales$\}$

Comments

Consider again the tableaux from Slides 537 and 541.

- both used only formulas of the form $P \to Q$ where P and Q are conjunctions,
  - forward reasoning: close left branch immediately,
  - backward reasoning: close right branch immediately,

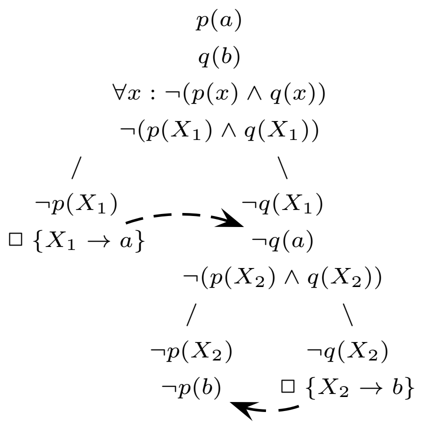
  $\Rightarrow$ linear proofs (if the correct rule is always chosen)

$\Rightarrow$ preview:
the resolution calculus provides an efficient calculus for such cases where only rules are used (Datalog)

- tableaux have higher expressiveness: can handle full disjunction etc.:
  Description Logics and OWL (Semantic Web) use tableaux

NON-CLOSED TABLEAUX: (TYPICAL) SAMPLE MODELS

Is the axiom $\forall x : \neg(p(x) \wedge q(x))$ together with the "database" $\{p(a),\ q(b)\}$ consistent?



- there is no way to close the tableau
- its non-closed path describes a model of the input formula
  (where $\neg q(a)$ and $\neg p(b)$ hold which are not specified in the database – open world reasoning)

Tableau Calculi: Application for Query Answering

Consider the database $\{\forall x : (p(x) \rightarrow q(x)),\ p(a), q(b)\}$ and the query $? - q(X)$.

$$\forall x : (p(x) \rightarrow q(x))$$
$$p(a)$$
$$q(b)$$
$\neg q(X)$ add the negated query with a free variable

- collect all substitutions of X that can be used to close the tableau.
- note: the substitution can comprise a the application of a Skolem function. Then, the "answer" can only be described as a thing that satisfies a certain existential formula.

  Consider $\forall x$: (person$(x) \rightarrow (\exists y$: person$(y) \wedge$ father$(x, y)))$,
  $\forall x, y, z$: ((father$(x, y) \wedge$ father$(y, z)) \rightarrow$ grandfather$(x, z))$,
  person(john), person(jack), father(john,jack) and the query ?- grandfather(john,X).

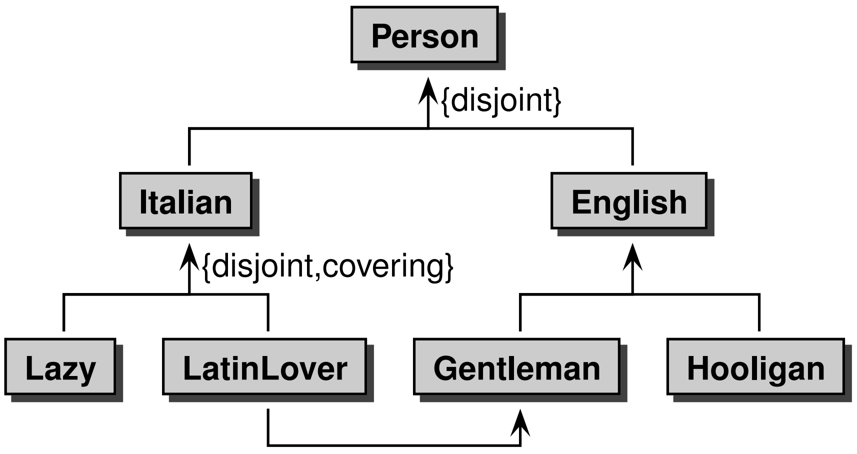
Tableau Calculi in General

- intuitive idea
- can be designed in this way for any logic (modal logics, description logics etc.)
- implementations use more efficient heuristics

Examples + Exercises

- prove that
  $$\forall x : ((p(x) \rightarrow q(x)) \wedge (q(x) \rightarrow r(x)))\ \models\ \forall x : (p(x) \rightarrow r(x))$$
  and
  $$\forall x : ((p(x) \rightarrow q(x)) \wedge (q(x) \rightarrow r(x)))\ \models\ (p(a) \rightarrow r(a))$$
- Consider the italian-vs-english ontology from Slide 546. Consider the statement "all Italians are lazy". Prove it or give a counterexample.
- Consider the italian-professors ontology from Slide 547. Is there anything interesting to prove?

Example: Italians and Englishmen



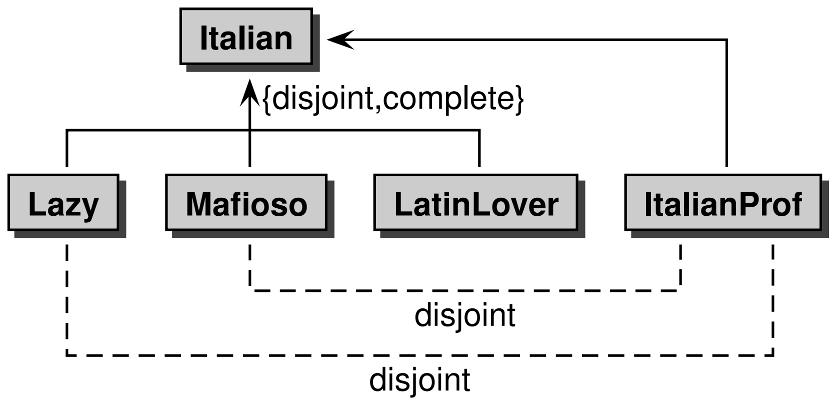
Exercise: write down as concise as possible *everything* that is implied by this ontology in text, set theory and first-order logic.

[by Enrico Franconi, REWERSE Summer School 2005]

[see Slide 548 for an excerpt and a relevant proof]

.

Example: Italian Professors



Exercise: write down as concise as possible *everything* that is implied by this ontology in text, set theory and first-order logic.

[by Enrico Franconi, REWERSE Summer School 2005]

Tableau Proof (Example)

Tableau for the italian-vs-english ontology from Slide 546 and the statement “all Italians are lazy”.

$$
\begin{array}{c}
\forall x : \text{italian}(x) \rightarrow \neg\text{english}(x)\ [1] \\
\forall x : \text{english}(x) \rightarrow \neg\text{italian}(x)\ [2] \\
\forall x : \text{italian}(x) \rightarrow (\text{lazy}(x) \vee \text{latinlover}(x))\ [3] \\
\forall x : \text{lazy}(x) \rightarrow \neg\text{latinlover}(x))\ [4] \\
\forall x : \text{latinlover}(x) \rightarrow \neg\text{lazy}(x)\ [5] \\
\forall x : \text{latinlover}(x) \rightarrow \text{gentleman}(x)\ [6] \\
\forall x : \text{gentleman}(x) \rightarrow \text{english}(x)\ [7] \\
\exists x : \text{italian}(x) \wedge \neg\text{lazy}(x)\ \ [8]\ \ \text{(negation of the claim)} \\
\mid\ \text{(skolemization of [8])} \\
\text{italian}(c) \wedge \neg\text{lazy}(c) \\
\text{italian}(c) \\
\neg\text{lazy}(c) \\
\mid\ \text{(use [3])} \\
\forall x : \text{italian}(x) \rightarrow (\text{lazy}(x) \vee \text{latinlover}(x)) \\
\text{italian}(X_1) \rightarrow (\text{lazy}(X_1) \vee \text{latinlover}(X_1))
\end{array}
$$

$$
\begin{array}{cc}
/ & \backslash \\
\neg\text{italian}(X_1) & \text{lazy}(X_1) \vee \text{latinlover}(X_1) \\
\Box\ \{X_1 \rightarrow c\} & \text{lazy}(c) \vee \text{latinlover}(c) \\
 & \begin{array}{cc} / & \backslash \\ \text{lazy}(c) & \text{latinlover}(c) \\ \Box & \end{array}
\end{array}
$$

Continue right branch using [6], [7] and finally [1] or [2].

Tableaux and Conjunctive Query Answering

“All organizations that have their headquarters in the capital of a European member country”

Using a simplified Mondial signature:

$$
\begin{aligned}
F(org) = \exists cty, ctry(&\text{headq}(org, cty, ctry) \wedge \text{capital}(ctry, cty) \wedge \\
&\text{enc}(ctry, \text{“Europe”}) \wedge \text{member}(ctry, org))
\end{aligned}
$$

Start the tableau with $\neg F(X)$.

$$\neg\exists cty, ctry(\text{headq}(org, cty, ctry) \wedge \text{capital}(ctry, cty) \wedge \text{enc}(ctry, \text{“Europe”}) \wedge \text{member}(ctry, org))$$

$$|$$

$$\neg(\text{headq}(Org, Cty, Ctry) \wedge \text{capital}(Ctry, Cty) \wedge \text{enc}(Ctry, \text{“Europe”}) \wedge \text{member}(Ctry, Org))$$

(β-rule + reordering)

$$/ \qquad\qquad / \qquad\qquad \backslash \qquad\qquad \backslash$$

$$\neg\text{enc}(Ctry, \text{“Europe”}) \quad \neg\text{capital}(Ctry, Cty) \quad \neg\text{headq}(Org, Cty, Ctry) \quad \neg\text{member}(Ctry, Org))$$

- Close left branch by “local answer set” $Ctry/$’D’, $Ctry/$’B’, etc.
- propagate answer set to 2nd branch, close for each $Ctry$ with appropriate value for Cty, e.g. $(Ctry/$’D’$, Cty/$’Berlin’$)$ and $(Ctry/$’B’$, Cty/$’Brussels’$)$.
- propagate answer sets to 3rd branch, close for each $(Ctry, Cty)$ with appropriate organization: $(Org/$’EU’$, Ctry/$’B’$, Cty/$’Brussels’$)$,
  $(Org/$’NATO’$, Ctry/$’B’$, Cty/$’Brussels’$)$, – no tuple for ’Berlin’.
- propagate answer sets to 4th branch, closes if $Ctry$ is a member of Org.

$\Rightarrow$ in this “simple” case (conjunctive query), the evaluation results in
$(\sigma[\text{Continent='Europe'}](\text{enc})) \bowtie \text{capital} \bowtie \text{headq} \ltimes \text{member}$

Exercise: do the same for “…of a European or Asian member country”.

Tableaux and $body \rightarrow head$ Rules

Consider again the tableaux from Slides 537 and 541.

- all used axioms are of the form $\forall x_1, \ldots, x_n : body \rightarrow head$,
- plus a (negated) query (“goal”) $\neg F(\bar{X})$,
- standard tableau pattern:

$$\forall x_1, \ldots, x_n : body(x_1, \ldots, x_n) \rightarrow head(x_1, \ldots, x_n)$$

$$body(X_1, \ldots, X_n) \rightarrow head(X_1, \ldots, X_n)$$

$$\neg body(X_1, \ldots, X_n) \vee head(X_1, \ldots, X_n)$$

$$/ \qquad\qquad\qquad \backslash$$

$$\neg body(X_1, \ldots, X_n) \qquad head(X_1, \ldots, X_n)$$

- close one branch immediately (forward: left, backward: right),
  obtain a set of tuples binding $(X_1, \ldots, X_n)$ (i.e., that satisfy $body$), propagate to the other branch,
- continue with next axiom.
- Again, the tableau is closed for all bindings of $\bar{X}$ that are answers.

Summary: Tableau Reasoning

- covers full first-order logic,
- theoretically incomplete,
- most practical cases result in acceptable performance,
- reasons for more complex tableaux:
  - search for proof tableaux/trees
  - disjunction (explore several branches where only one contributes)
  - multiple instantiations of universally quantified variables
    - needed for self-joins, transitivity,
    - especially in combination with skolemized $\exists$-terms.
- simple patterns (rules, conjunctive body/head) result in effectively nearly-algebraic evaluation.
- But: for simple patterns one does not need a full first-order reasoner.

Properties of First-Order Logic Decision Procedures

- calculi (=algorithms) for checking if $F \models G$
  (often by proving that $F \wedge \neg G$ is unsatisfiable)
- write $F \vdash_C G$ if calculus C proves that $F \models G$.
- Correctness of a calculus: $F \vdash_C G \Rightarrow F \models G$
- Completeness of a calculus: $F \models G \Rightarrow F \vdash_C G$
- there are complete calculi and proof procedures for propositional logic (e.g., Tableau Calculus or Model Checking)
- if a logic is undecidable (like first-order logic) then there cannot be any complete calculus!

What to do?

$\Rightarrow$ use an undecidable logic and a correct, but (theoretically) incomplete calculus.

- e.g. Software Verification.

$\Rightarrow$ use a decidable logic (i.e., weaker than FOL).

- often a restricted set of formulas (Description Logic [Semantic Web], Datalog Variants [Database Theory])

ASIDE: WHY “FIRST-ORDER”-LOGIC?

Recall:

- there is a domain $\mathcal{D}$. Functions and precidates talk *about* elements of $\mathcal{D}$.
- there is no way to talk *about* functions or predicates.

Higher-Order-Logics

- the elements of the domain $\mathcal{D}$ are “first-order things”
- sets, functions and predicates are “second-order things”
- predicates about predicates are higher-order things
- higher-order logics can be used for reasoning *about* metadata

Example

- Transitivity as a property of predicates is second order:
  $\forall p : \text{transitive}(p) \rightarrow (\forall x, z : (\exists y : (p(x,y) \wedge p(y,z)) \rightarrow p(x,z)))$
  Note that transitivity of *a certain* predicate is first-order:
  $\forall x, z : ((\exists y : (\text{ancestor}(x,y) \wedge \text{ancestor}(y,z))) \rightarrow \text{ancestor}(x,z))$

Aside: Induction Axiom as Example for Second Order Logic

- a well-founded domain d (i.e., a finite set of minimal elements (for which min(d,x) holds) from which the domain can be enumerated by a successor predicate
  (Natural numbers: 1, succ(i,i+1))
- well-founded: unary 2nd-order predicate over sets
- The induction axiom as a 2nd order logic formula:

$$\begin{aligned}\forall p, d : \quad & (\text{well-founded}(d) \wedge (\forall x : \text{min}(d,x) \rightarrow p(x)) \wedge (\forall x, y : p(x) \wedge \text{succ}(x,y) \rightarrow p(y))) \rightarrow \\ & (\forall x : d(x) \rightarrow p(x))\end{aligned}$$

For natural numbers:

$$\forall p : (p(1) \wedge (\forall x : p(x) \rightarrow p(x+1))) \rightarrow (\forall x \in \mathbb{N} : p(x))$$

Aside: Paradoxes can be formulated in 2nd Order Logic

"X is the set of all sets that do not contain themselves"

$$X = \{z : z \notin z\}$$

A set "is" a unary predicate: $X(z)$ holds if z is an element of X
(for example, classes, i.e., Person(x), City(x))

Logical characterization of X: $X(z) \leftrightarrow \neg z(z)$,

applied to $z := X$ – is X in X? $X(X) \leftrightarrow \neg X(X)$.

... can neither be true nor false.

How to avoid paradoxes

Paradoxes can be avoided if each variable *either* ranges over first-order things (elements of the domain) or over second-order things (predicates).