WWU MÜNSTER

Foundations: Logic

Statistical Relational Artificial Intelligence (StaRAI)

$$\neg(p \land (q \Rightarrow r))$$
$$\neg(p \land (\neg q \lor r))$$
$$\neg p \lor \neg(\neg q \lor r)$$
$$\neg p \lor (\neg\neg q \land \neg r)$$
$$\neg p \lor (q \land \neg r)$$

p	q	$p \land q$
T	T	T
T	F	F
F	T	F
F	F	F

$$knows(jack, jill)$$
$$\land$$
$$\forall X, Y, X \in \mathrm{D} \land Y \in \mathrm{D}.$$
$$\big(knows(X, Y) \Rightarrow knows(Y, X)\big)$$

Contents

Overview: 2. Foundations

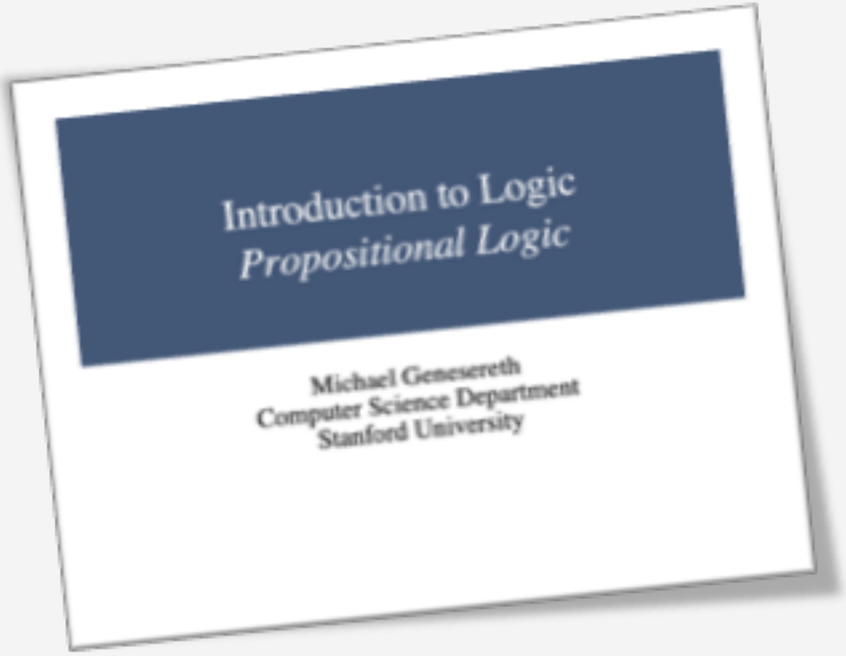
A. *Logic*
- Propositional logic: alphabet, grammar, normal forms, rules
- First-order logic: introducing quantifiers, domain constraints

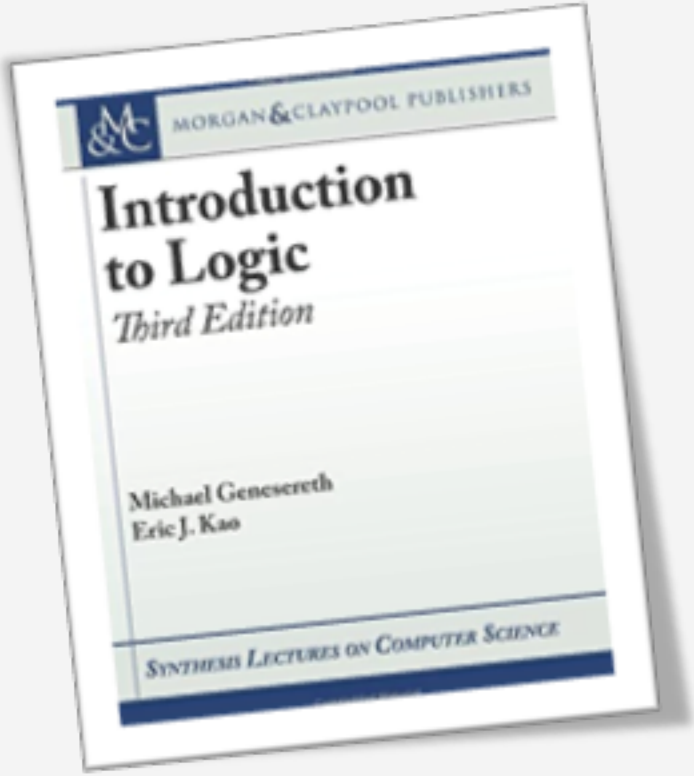
B. *Probability theory*
- Modelling: (conditional) probability distributions, random variables, marginal and joint distributions
- Inference: axioms and basic rules, Bayes theorem, independence

C. *Probabilistic graphical models*
- Syntax, semantics
- Inference problems

Sources

- Slides based on the lecture “Introduction to Logic” by Michael Genesereth at Stanford University, which has a book: “Introduction to Logic” by Michael Genesereth & Eric J. Kao





Logic

- Logic for natural language modelling, hardware + software engineering, games etc.
- Describe constraints about a world
- Components
  - Logical language

    Andy likes Cody. *Abby does **not** like Dana.* *Bess likes **everyone** that Bess likes.*
    *Dana does **not** like Abby.* *Bess likes Cody **or** Dana.*

  - Logical reasoning

Premises:	**Truths:**	**Falsehoods:**	**Unknowns:**
Dana likes Cody.	*Bess likes Cody.*	*Bess likes Dana.*	*Dana likes Bess.*
Abby does not like Dana.	*Bess does not like Dana.*	*Everybody likes everybody.*	
Everybody likes somebody.	*Everybody likes someone.*		
Bess likes Cody or Dana.			
Abby likes everyone that Bess likes.			
Cody likes everyone who likes her.			
Nobody likes herself.			

- Rules of inference
- Soundness and completeness
- Model checking (enumeration of cases)

Syntax

- Propositional vocabulary:
  - Set of of primitive symbols called *proposition constants*
  - Convention: strings of alphanumeric characters, starting lowercase
    - $raining, umbrella, sick, parent$
    - p, q, r
    - $parent1, parent2, r32aining, rAiNiNg$
    - Wrong: $4815162342, rain.snowing$
- Propositional sentence
  - Member of propositional vocabulary
  - Compound expression: combination of
    - Member of vocabulary
    - Logical operators ($\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$)
    - Parentheses
  - Subexpressions sometimes referred to using Greek lowercase letters: ϕ, ψ
- Propositional language
  - Set of all propositional sentences that can be formed from a propositional vocabulary

Syntax

- Compound sentences
  - Negation: $\neg raining$
    - Argument called *target*
  - Conjunction: $raining \land umbrella$
    - Arguments called *conjuncts*
  - Disjunction: $raining \lor umbrella$
    - Arguments called *disjuncts*
  - Implication: $raining \Rightarrow umbrella$
  - Equivalence: $raining \Leftrightarrow umbrella$
  - Further nested sentences
    - $\neg\, raining \lor umbrella$
    - $\neg\, (raining \land umbrella)$

- Precedence: $\neg, \land, \lor, \Rightarrow, \Leftrightarrow$
  - Allows for omitting parentheses
    - $\neg p \lor q \equiv \big((\neg p) \lor q\big)$
    - $p \lor q \land r \equiv \big(p \lor (q \land r)\big)$
    - $p \lor q \Rightarrow r \equiv \big((p \lor q) \Rightarrow r\big)$
    - $p \Rightarrow q \Leftrightarrow r \equiv \big((p \Rightarrow q) \Leftrightarrow r\big)$
  - Association
    - $\land, \lor$ *left-associative*
      - $p \land q \land r \equiv \big((p \land q) \land r\big)$
    - $\Rightarrow, \Leftrightarrow$ *right-associative*
      - $p \Rightarrow q \Rightarrow r \equiv \big(p \Rightarrow (q \Rightarrow r)\big)$

Semantics

- Propositional interpretation (*possible world*)
  - Association between propositional constants in propositional language and *truth values* T or F

$$p \xrightarrow{i} \mathrm{T} \qquad p^i = \mathrm{T}$$
$$q \xrightarrow{i} \mathrm{F} \qquad q^i = \mathrm{F}$$
$$r \xrightarrow{i} \mathrm{T} \qquad r^i = \mathrm{T}$$

  - Sometimes considered a Boolean vector of values for items in the ordered signature of the language
    - Ordered signature: constants of the language in a given order
    - Order: pqr
    - Interpretation $i = \mathrm{TFT}$
  - Sometimes T or F denoted by 1,0

Semantics

One interpretation is as good as any other in absence of additional information.

- Sentential interpretation
  - Association between sentences in a propositional language and truth values T or F
  - Propositional interpretation defines sentential interpretation by applying operator semantics
    - *Operator semantics*

ϕ	$\neg\phi$
T	F
F	T

ϕ	ψ	$\phi \wedge \psi$
T	T	T
T	F	F
F	T	F
F	F	F

ϕ	ψ	$\phi \vee \psi$
T	T	T
T	F	T
F	T	T
F	F	F

ϕ	ψ	$\phi \Rightarrow \psi$
T	T	T
T	F	F
F	T	T
F	F	T

ϕ	ψ	$\phi \Leftrightarrow \psi$
T	T	T
T	F	F
F	T	F
F	F	T

  - *Evaluation*:

$$p^i = \mathrm{T} \qquad (p \vee q)^i = (\mathrm{T} \vee \mathrm{F}) = \mathrm{T}$$

$$q^i = \mathrm{F} \qquad (\neg q \vee r)^i = (\neg \mathrm{F} \vee \mathrm{T}) = (\mathrm{T} \vee \mathrm{T}) = \mathrm{T}$$

$$r^i = \mathrm{T} \qquad \left((p \vee q) \wedge (\neg q \vee r)\right)^i = (\mathrm{T} \wedge \mathrm{T}) = \mathrm{T}$$

Semantics

- Truth table
  - Table of all possible interpretations for the propositional constants in a language
    - One column per constant
    - One row per interpretation
    - For a language with n constants, there are 2^n interpretations
      - Exponential!

p	q	r
T	T	T
T	T	F
T	F	T
T	F	F
F	T	T
F	T	F
F	F	T
F	F	F

Satisfaction (Model Checking)

Proofs (symbolic manipulation of sentences) usually smaller than truth tables and thus often less work.

- Find all propositional interpretations that satisfy a given set of sentences
  - Satisfy: each sentence evaluates to T
  - Satisfying interpretation then often called *model*
  - Procedure
    1. Form a truth table for propositional constants
    2. For each sentence in the set and each row in the truth table, check whether the row satisfies the sentence
       - Cross out rows that do not
    - Result: remaining rows, which satisfy all sentences in the given set
  - Example
    - $q \Rightarrow r$
    - $p \Rightarrow q \land r$
    - $\neg r$

p	q	r
~~T~~	~~T~~	~~T~~
~~T~~	~~T~~	~~F~~
~~T~~	~~F~~	~~T~~
~~T~~	~~F~~	~~F~~
~~F~~	~~T~~	~~T~~
~~F~~	~~T~~	~~F~~
~~F~~	~~F~~	~~T~~
F	F	F

model

Satisfaction: Properties of Sentences

- Valid sentence
  - If and only if *every* interpretation satisfies it
- Contingent sentence
  - If and only if *some* interpretation satisfies it and *some* interpretation falsifies it
- Unsatisfiable sentence
  - If and only if *no* interpretation satisfies it

- Satisfiable
  - If and only if it is *either valid or contingent*
- Falsifiable
  - If and only if it is *either contingent or unsatisfiable*

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

p	q	$p \vee q \vee \neg(p \wedge q)$
T	T	T
T	F	T
F	T	T
F	F	T

Which properties hold?

Satisfaction: Logical Equivalence

- Logical equivalence of sentences ϕ, ψ
  - If and only if every truth assignment that satisfies ϕ satisfies ψ *and* every truth assignment that satisfies ψ satisfies ϕ
  - Common equivalences:
    - Double negation: $p \Leftrightarrow \neg\neg p$
    - de Morgan's laws
      - $\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$
      - $\neg(p \vee q) \Leftrightarrow (\neg p \wedge \neg q)$
    - Implication: $(p \Rightarrow q) \Leftrightarrow (\neg p \vee q)$
    - Equivalence: $(p \Leftrightarrow q) \Leftrightarrow \big((p \Rightarrow q) \wedge (q \Rightarrow p)\big)$
    - Distributive laws
      - $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$
      - $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$

Satisfaction: Logical Entailment

- Logical entailment $\phi \vDash \psi$
  - Premise ϕ logically entails conclusion if and only if every interpretation that satisfies ϕ also satisfies ψ
    - Satisfying interpretations of ϕ need to be a subset of the satisfying interpretations of ψ
  - For sets: A set of premises entails a set of conclusions if and only if every interpretation that satisfies all premises also satisfies all conclusions
  - Examples
    - $(p \land q) \vDash (p \lor q)$
    - $p \vDash (p \lor q)$
    - $(p \land q) \vDash p$
    - $p \nvDash (p \land q)$

p	q	$p \land q$
T	T	T
T	F	F
F	T	F
F	F	F

p	q	$p \lor q$
T	T	T
T	F	T
F	T	T
F	F	F

Satisfaction: Consistency

- Consistency of sentences ϕ, ψ
  - If and only there is a truth assignment that satisfies both ϕ and ψ
  - Examples
    - p is logically consistent with q
    - $(p \lor q)$ is logically consistent with $(\neg p \lor \neg q)$
    - $(p \Rightarrow q)$ is logically consistent with $(\neg p \lor q)$
    - p is not consistent with $\neg p$

p	q	$p \land q$
T	T	T
T	F	F
F	T	F
F	F	F

p	q	$p \lor q$
T	T	T
T	F	T
F	T	T
F	F	F

Some Connections between These Concepts

- Equivalence Theorem
  - Sentences ϕ, ψ are logically equivalent if and only if $(\phi \Leftrightarrow \psi)$ is valid
- Unsatisfiability Theorem
  - $\Delta \vDash \psi$ if and only if $\Delta \cup \{\neg\psi\}$ is unsatisfiable
- Deduction Theorem
  - Sentence ϕ logically entails sentence ψ if and only if $(\phi \Rightarrow \psi)$ is valid
- Consistency Theorem
  - Sentence ϕ logically consistent with sentence ψ if and only if $(\phi \wedge \psi)$ is satisfiable

Normal Forms (NFs)

- *Literal*: either a proposition constant or its negation
  - $p, \neg p$
- Clause: either a literal or a disjunction of literals
  - $p, \neg p, \neg p \vee q$
- Negation normal form (NNF)
  - Sentences consisting of disjunctions and conjunctions of literals
    - p
    - $p \wedge q$
    - $(\neg p \wedge q) \vee \big(p \wedge (r \vee s)\big)$
- Conjunctive normal form (CNF)
  - Conjunction of clauses
    - p (single conjunct)
    - $(\neg p \vee q)$ (single conjunct)
    - $p \wedge q$
    - $(\neg p \vee q) \wedge (p \vee r) \wedge s$
- Disjunctive normal form (DNF)
  - Disjunction of conjunctions of literals
    - p (single disjunct)
    - $(p \wedge q)$ (single disjunct)
    - $\neg p \vee q$
    - $(\neg p \wedge q) \vee (p \wedge r) \vee s$

Normal Forms (NFs): Conversion into NNF

- Given a set of sentences
- For each sentence:
  1. Equivalences $\phi \Leftrightarrow \psi$ out
     - Replace with $(\phi \Rightarrow \psi) \wedge (q \Rightarrow \psi)$
  2. Implications $\phi \Rightarrow \psi$ out
     - Replace with $(\neg\phi \vee \psi)$
  3. Negations in
     - Replace $\neg\neg p$ with p
     - Apply de Morgan's laws
       - $\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$
       - $\neg(p \vee q) \Leftrightarrow (\neg p \wedge \neg q)$
- Result: sentences in NNF

- Example

$$\neg\big(p \wedge (q \Rightarrow r)\big)$$
$$\neg\big(p \wedge (\neg q \vee r)\big)$$
$$\neg p \vee \neg(\neg q \vee r)$$
$$\neg p \vee (\neg\neg q \wedge \neg r)$$
$$\neg p \vee (q \wedge \neg r)$$

  - In NNF

Normal Forms (NFs): Conversion into CNF / DNF

1. Convert into NNF
2. Distribute depending on NF
   - CNF: … disjunctions inward
     - Replace $\phi_1 \vee (\phi_2 \wedge \phi_3)$ with $(\phi_1 \vee \phi_2) \wedge (\phi_1 \vee \phi_3)$
   - DNF: … conjunctions inwards
     - Replace $\phi_1 \wedge (\phi_2 \vee \phi_3)$ with $(\phi_1 \wedge \phi_2) \vee (\phi_1 \wedge \phi_3)$

- Example

$$\neg(p \wedge (q \Rightarrow r))$$
$$\neg(p \wedge (\neg q \vee r))$$
$$\neg p \vee \neg(\neg q \vee r)$$
$$\neg p \vee (\neg\neg q \wedge \neg r)$$
$$\neg p \vee (q \wedge \neg r)$$

  - In NNF

What about CNF / DNF?

- Already in DNF
- To get a CNF, distribute $\neg p \vee$ inwards: $(\neg p \vee q) \wedge (\neg p \vee \neg r)$

Interim Summary

- Syntax
  - Propositional vocabulary of proposition constants
  - Propositional sentence: combination of proposition constants, logical operators, parenthesis
  - Logical operators: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$
- Semantics
  - Propositional interpretations: truth assignments to proposition constants
  - Sentential interpretations: evaluations of sentences given propositional interpretation
  - Satisfaction: all propositional interpretations that satisfy all sentences
    - Properties: valid, contingent, unsatisfiable; satisfiable, falsifiable
    - Comparison of sentences: equivalence, entailment, consistency; theorems for connections
- Normal forms
  - NNF, CNF, DNF

First-order Logic with Domain Constraints

Introducing quantifiers and domain constraints

* Based on "Lifted Inference and Learning in Statistical Relational Models" by Guy Van den Broeck, 2013
→ Some notions are used a bit differently than in some classical logic literature but I want to keep the slides consistent with this source

Propositional → First-order Logic (FOL)

- Motivation: Being able to talk about objects and relations among them
  - Example: in propositional logic
    - Premises
      - If Jack knows Jill, then Jill knows Jack.
      - Jack knows Jill.
    - Conclusion: Does Jill know Jack? (Yes.)
  - Example: in first-order (or relational) logic
    - Premises
      - If one person knows another person, then the other person knows the first person.
      - Jack knows Jill.
    - Conclusion: Does Jill know Jack? (Yes.)

- New linguistic features
  - (Logical) variables
    - X, Y
  - Quantifiers ∀, ∃
    - Followed by a logical variable
  - Example
    - Premises
      - $\forall X. \forall Y. \left(knows(X, Y) \Rightarrow knows(Y, X)\right)$
      - $knows(jack, jill)$
    - Conclusion: $knows(jill, jack)$?

FOL with Domain Constraints (FOL-DC)

- FOL has a domain of discourse that is possibly infinite
  - Quantifiers can range over infinite sets of objects
- Domain constraints
  - Restrict quantifiers to finite domains
- Why?
  - For probabilistic inference, infinity is hard to handle
    - Restriction allows for obtaining a well-defined probability distribution over the ground terms
    - Enables weighted model counting as a way to answer queries about probability (distributions)
  - Tasks like checking validity or satisfiability become decidable in FOL-DC
    - Undecidable in full FOL
    - Not the only decidable subclass that exists, but the one that we need for the lecture

Syntax: *Function-free* FOL

- Logical symbols
  - Logical connectives ¬, ∧, ∨, etc.
  - Quantifiers ∀, ∃
    - *Universal* quantifier ∀ (*for all*)
    - *Existence* quantifier ∃ (*there exists*)
    - Mostly, we will be dealing with ∀ later on
  - Truth values T and F
  - Set of logical variables $\boldsymbol{X} = \{X, Y, \dots\}$
    - Usually from the end of the alphabet
  - (Parentheses and punctuation)
- Non-logical symbols
  - Set of predicate symbols p/n of *arity* n
    - Including propositional variables with arity $n = 0$
    - Sometimes called relation constants
  - A set of (proposition) constant symbols $\{a, b, c, \dots\}$
    - Alphanumeric, usually starting with a letter from the beginning of the alphabet; numbers now also allowed
    - Sometimes called object constants
    - With a natural order on the constants

Syntax: *Function-free* FOL

- Signature: set of constants together with a set of predicates including arity
- Sentences no longer of one type
  - C.f., propositional logic
  - *Relational* sentences
    - n-ary predicate with n logical variables or constants in parentheses and with commas
  - *Logical* sentences
    - Relational sentences possibly combined with logical operators
  - *Quantified* sentences
    - Logical sentences that contain quantifiers for some or all logical variables occurring

- Examples
  - Signature
    - Constants: $jack, jill$
    - Predicate: $knows/2, raining/0$
  - Logical variables: X, Y
  - Relational sentence
    - $knows(X, Y)$
    - $knows(jack, X)$
  - Logical sentence
    - $knows(X, Y) \Rightarrow knows(Y, X)$
  - Quantified sentence
    - $\forall X. \forall Y. \left(knows(X, Y) \Rightarrow knows(Y, X)\right)$

Syntax: *Function-free* FOL-DC

- Function-free FOL extended with
  - Logical symbols
    - Less-than $<$
    - Set membership and inclusion $\in, \subseteq$
    - Set of domain variables $\mathbf{D} = \{\mathrm{D}, \mathrm{F}, \dots\}$
    - Set operations $\cup, \cap, \setminus$
  - Non-logical symbols (signature)
    - Set of "sets of constants" symbols $\{\{a, b, c\}, \{a, d\}, \emptyset, \dots\}$

- These constructs only occur when restricting domains using so-called constraint sets

Syntax: FOL-DC – Grammar

- Logical term
  - Logical variable or constant
  - Refers to objects in the domain of discourse (scenario)
  - Examples
    - Logical variables: X, Y
    - Constants: $jack, jill$

- Logical atom (*relational sentence*)
  - $p(t_1, \ldots, t_n)$
  - Apply predicate p/n to n-tuple of logical term arguments t_i
  - Examples
    - Predicate with logical variables only as arguments: $knows(X, Y)$
    - Predicate with a logical variable and a constant as arguments: $knows(jack, X)$
    - Predicate with constants only as arguments: $knows(jack, jill)$
    - 0-ary predicate (propositional atom): $raining$

Syntax: FOL-DC – Grammar

- Domain term
  - Domain variable, set of constants, or combination of two other domain terms t_d, t'_d in the form of
    $$t_d \cup t'_d, t_d \cap t'_d, t_d \setminus t'_d$$
  - Refers to sets of objects in the domain of discourse
  - Examples
    - Domain variables: D, E
    - Sets of constants: $\{a, b, c\}, \{jill, jack\}$
    - Combination of domain terms: $\mathrm{D} \cup \mathrm{E}, \mathrm{D} \cup \{jill, jack\}$

- Domain atom
  - Between logical terms t_l, t'_l
    $$t_l = t'_l$$
    $$t_l < t'_l$$
    - $X = jill$
    - $X < Y$ (natural order on constants needed)
  - Between domains terms t_d, t'_d
    $$t_d = t'_d$$
    $$t_d \subseteq t'_d$$
    - $\mathrm{D} = \mathrm{E}, \mathrm{E} \subseteq \mathrm{D} \cup \{jill, jack\}$
  - Between logical terms t_l, domain term t_d
    $$t_l \in t_d$$
    - $X \in \{jill, jack\}$

Syntax: FOL-DC – Grammar

- Domain constraint
  - Domain atom or its negation ($\neq, \notin, \nsubseteq$)
- Constraint set
  - Conjunction of domain constraints
    - (restricted to conjunction for the sake of simplicity)
    - (more expressive constraint languages would have the purpose to express certain constraint sets much more precisely)
- Example
  - $X \in \mathrm{Bird} \land X \neq kiwi$

Syntax: FOL-DC – Grammar

- (Well-formed) formula
  - Logical atom
  - If φ, ψ formulas, then the following are
    - $\neg\varphi$
    - *Extensional conjunction* $\varphi \wedge \psi$ or
    - *Extensional disjunction* $\varphi \vee \psi$
  - If φ formula, $\boldsymbol{V}$ a (possibly empty) set of (logical or domain) variables, and cs a constraint set that contains at least one domain atom of the form $V = t$, $V \in t$, or $V \subseteq t$ for every $V \in \boldsymbol{V}$, the following are
    - *Intensional conjunction* $\forall \boldsymbol{V}, cs : \varphi$
    - *Intensional disjunction* $\exists \boldsymbol{V}, cs : \varphi$
- A variable is bound
  - If it is quantified by an enclosing intensional conjunction or disjunction
- A variable is free
  - If it is not bound
- Sentence
  - Formula without free variables
- Formula is ground
  - If it does not contain any variables

Syntax: FOL-DC – Grammar

- Examples:
  - Domain atom:
    - $\mathrm{D} = \{jack, jill\}$
  - Logical atom:
    - $knows(X, Y), knows(jack, jill)$
  - *Extensional disjunction:*
    - $raining \vee snowing$
  - *Intensional conjunction:*
    $$\forall X, Y, X \in \mathrm{D} \wedge Y \in \mathrm{D}.\ \big(knows(X, Y) \Rightarrow knows(Y, X)\big)$$
    - Logical variables: X, Y
    - Constraint set: $X \in \mathrm{D} \wedge Y \in \mathrm{D}$

- Consider formula
  $$\forall X, X \in \mathrm{D}.\ \big(knows(X, Y) \Rightarrow knows(Y, X)\big)$$
  - Bound variable: X
  - Free variable: Y
    - Essentially universally quantified
  - Not a sentence because Y is free
    - Sentence:
      $$\forall X, Y, X \in \mathrm{D} \wedge Y \in \mathrm{D}.\ \big(knows(X, Y) \Rightarrow knows(Y, X)\big)$$
  - Ground:
    - $knows(jack, jill) \Rightarrow knows(jill, jack)$
    - $raining \vee snowing$

Some More Information about Quantifiers

- Quantifiers can be nested
  - $\forall X. \left(\exists Y. knows(X, Y)\right)$
- Precedence: quantifiers have higher precedence than logical operators
- Quantifier reversal
  - $\forall X. \forall Y. p(X, Y) \Leftrightarrow \forall Y. \forall X. p(X, Y)$
  - $\exists X. \exists Y. p(X, Y) \Leftrightarrow \exists Y. \exists X. p(X, Y)$
- Existential distribution
  - $\exists Y. \forall X. p(X, Y) \Rightarrow \forall X. \exists Y. p(X, Y)$
- Negation distribution
  - $\neg \forall X. \phi \Leftrightarrow \exists X. \neg \phi$
  - $\neg \exists X. \phi \Leftrightarrow \forall X. \neg \phi$

Further Terminology

- Literal l
  - Logical atom a or its negation $\neg a$
- Clause
  - Disjunction of literals
- Theory in conjunctive normal form (CNF)
  - Conjunction of clauses
- Term
  - Conjunction of literals
- Theory in disjunctive normal form (DNF)
  - Disjunction of terms
- Theory or knowledge base
  - Conjunction of formulas

- Examples
  - Literals
    - $knows(X, Y), \neg knows(X, Y), raining$
  - Clause
    - $raining \vee knows(jill, Y)$
  - Term
    - $knows(X, Y) \wedge knows(Y, X)$
  - Knowledge base

$$\begin{gathered} knows(jack, jill) \\ \wedge \\ \forall X, Y, X \in \mathrm{D} \wedge Y \in \mathrm{D}. \\ \big(knows(X, Y) \Rightarrow knows(Y, X)\big) \end{gathered}$$

Semantics: Herbrand Base

- Herbrand base
  - Set of all grounded logical atoms that can be formed
  - Leads to a propositional language
- Instance
  - Consistently replace every occurrence of a free variable with a constant
    - *Consistent replacement*: If one occurrence of a variable replaced by a constant, then all occurrences of that variable replaced by the same constant

- Example
  - Herbrand base
    - Constants: $jack, jill$
    - Predicate: $knows/2, raining/0$
    - $knows(jack, jill), knows(jack, jack),$ $knows(jill, jack). knows(jill, jill), raining$
  - Example instances
    - $knows(X, Y)$
      - $knows(jack, jill)$
      - $knows(jill, jill)$
    - $p(X) \Rightarrow q(X)$
      - $p(a) \Rightarrow q(a)$

Semantics: Truth Assignment

Evaluation and satisfaction then work as before. Thus, properties may also hold or not and formulas may be compared (equivalence, entailment, consistency).

- Truth assignments to a Herbrand base works as in propositional languages
  - Propositional truth assignments
  - Sentential truth assignments
    - Same operator semantics
  - Universally quantified formula true for a truth assignment if and only if every instance of the scope of the quantified formula is true for that assignment
  - Existentially quantified formula is true for a truth assignment if and only if some instance of the scope of the quantified sentence is true for that assignment

- Example
  - Formula
    $$\forall X, Y, X \in \mathrm{D} \land Y \in \mathrm{D}. \left(knows(X, Y) \Rightarrow knows(Y, X)\right)$$
  - $knows(jack, jill), knows(jack, jack), knows(jill, jack). knows(jill, jill)$
  - Truth assignment TTTT
    - Formula true
  - Truth assignment FTTF
    - Formula true
  - Truth assignment FTTT
    - Formula false

Interim Summary

- Function-free first-order logic with domain constraints contains new constructs
  - Logical variables
  - Quantifiers
  - Domain variables and domain constraints
  - Quantifier equivalences
- Semantics
  - Herbrand base: grounding
  - Interpretations work as before
    - Quantifiers semantics
    - Evaluation and satisfaction work as before
      - Properties, comparison as well

Overview: 2. Foundations

A. *Logic*
- Propositional logic: alphabet, grammar, normal forms, rules
- First-order logic: introducing quantifiers, domain constraints

B. ***Probability theory***
- Modelling: (conditional) probability distributions, random variables, marginal and joint distributions
- Inference: axioms and basic rules, Bayes theorem, independence

C. *Probabilistic graphical models*
- Syntax, semantics
- Inference problems