CS-E4800 Artificial Intelligence

Jussi Rintanen

Department of Computer Science
Aalto University

February 6, 2020

Today's Lecture

- Predicate logic
- Logic for knowledge representation and reasoning
- Logic in natural language semantics
- Logic vs. other reasoning and problem-solving frameworks

The Zebra Puzzle

There are five houses.
The Englishman lives in the red house.
The Spaniard owns the dog.
Coffee is drunk in the green house.
The Ukrainian drinks tea.
The green house is immediately to the right of the ivory house.
The Old Gold smoker owns snails.
Kools are smoked in the yellow house.
Milk is drunk in the middle house.
The Norwegian lives in the first house.
The man who smokes Chesterfields lives in the house next to the man with the fox.
Kools are smoked in the house next to the house where the horse is kept.
The Lucky Strike smoker drinks orange juice.
The Japanese smokes Parliaments.
The Norwegian lives next to the blue house.

The Zebra Puzzle: Implicit Assumptions

Not all assumptions of the puzzle are explicit:

1. Each gentleman has exactly one drink, pet, house and cigarette.
2. Each drink, pet, house, and cigarette belongs to exactly one gentleman.
3. The houses are in a row (1st, 2nd, 3rd (middle), 4th, 5th)

All assumptions have to be made explicit!

The Zebra Puzzle

The atomic formulas needed for formalizing the puzzle (e.g. $(\mathit{LivesIn}(\mathit{Eng}, 2))$ are constructed with the help of the following object names.

$H = \{1, 2, 3, 4, 5\}$
$M = \{\mathrm{Eng}, \mathrm{Spa}, \mathrm{Ukr}, \mathrm{Nor}, \mathrm{Jap}\}$
$C = \{\mathrm{OldGold}, \mathrm{Kools}, \mathrm{Chesterfields}, \mathrm{LuckyStrike}, \mathrm{Parliaments}\}$
$P = \{\mathrm{Red}, \mathrm{Green}, \mathrm{Ivory}, \mathrm{Yellow}, \mathrm{Blue}\}$
$A = \{\mathrm{Dog}, \mathrm{Snails}, \mathrm{Fox}, \mathrm{Horse}, \mathrm{Zebra}\}$
$D = \{\mathrm{Coffee}, \mathrm{Tea}, \mathrm{Milk}, \mathrm{OrangeJuice}, \mathrm{Water}\}$

We use the same kind of schematic representation we saw in connection with NDL.

The Zebra Puzzle

The Englishman lives in the red house:
$\bigvee_{i \in H}(\mathit{LivesIn}(\mathit{Eng}, i) \wedge \mathit{Color}(i, \mathit{Red}))$

The Spaniard owns the dog: $\mathit{Pet}(\mathit{Spa}, \mathit{Dog})$

Coffee is drunk in the green house:
$\bigvee_{i \in M} \bigvee_{j \in H}(\mathit{LivesIn}(i, j) \wedge \mathit{Drinks}(i, \mathit{Coffee}) \wedge \mathit{Color}(j, \mathit{Green}))$

The Ukrainian drinks tea: $\mathit{Drinks}(\mathit{Ukr}, \mathit{Tea})$

The green house is to the right of the ivory house:
$\bigvee_{i \in H} \bigvee_{j \in H}(\mathit{Color}(i, \mathit{Ivory}) \wedge \mathit{Color}(j, \mathit{Green}) \wedge (j = i + 1))$

The Old Gold smoker owns snails:
$\bigvee_{i \in M}(\mathit{Smokes}(i, \mathit{OldGold}) \wedge \mathit{Pet}(i, \mathit{Snails}))$

The Zebra Puzzle

The Englishman lives in the red house:
$\bigvee_{i \in H}(\mathit{LivesIn}(\mathit{Eng}, i) \wedge \mathit{Color}(i, \mathit{Red}))$

Written in propositional logic in full:

$$
\begin{aligned}
&(\mathit{LivesIn}(\mathit{Eng}, 1) \wedge \mathit{Color}(1, \mathit{Red})) \\
&\vee(\mathit{LivesIn}(\mathit{Eng}, 2) \wedge \mathit{Color}(2, \mathit{Red})) \\
&\vee(\mathit{LivesIn}(\mathit{Eng}, 3) \wedge \mathit{Color}(3, \mathit{Red})) \\
&\vee(\mathit{LivesIn}(\mathit{Eng}, 4) \wedge \mathit{Color}(4, \mathit{Red})) \\
&\vee(\mathit{LivesIn}(\mathit{Eng}, 5) \wedge \mathit{Color}(5, \mathit{Red}))
\end{aligned}
$$

The Zebra Puzzle

The green house is to the right of the ivory house:
$\bigvee_{i \in H} \bigvee_{j \in H} (\mathit{Color}(i, \mathit{Ivory}) \wedge \mathit{Color}(j, \mathit{Green}) \wedge (j = i + 1))$

Written in propositional logic:

$$
\begin{aligned}
&(\mathit{Color}(1, \mathit{Ivory}) \wedge \mathit{Color}(2, \mathit{Green})) \\
&\vee (\mathit{Color}(2, \mathit{Ivory}) \wedge \mathit{Color}(3, \mathit{Green})) \\
&\vee (\mathit{Color}(3, \mathit{Ivory}) \wedge \mathit{Color}(4, \mathit{Green})) \\
&\vee (\mathit{Color}(4, \mathit{Ivory}) \wedge \mathit{Color}(5, \mathit{Green}))
\end{aligned}
$$

Here e.g. Color(1,Ivory) is understood as a name "Color(1,Ivory)" of an atomic proposition, without structure/content.

The Zebra Puzzle

Kools are smoked in the yellow house:
$\bigvee_{i \in M} \bigvee_{j \in H} (\mathit{LivesIn}(i, j) \wedge \mathit{Color}(j, \mathit{Yellow}) \wedge \mathit{Smokes}(i, \mathit{Kools}))$

Milk is drunk in the middle house:
$\bigvee_{i \in M} (\mathit{LivesIn}(i, 3) \wedge \mathit{Drinks}(i, \mathit{Milk}))$

The Norwegian lives in the first house: $\mathit{LivesIn}(\mathit{Nor}, 1)$

The man who smokes Chesterfields lives in the house next to the man with the fox: $\bigvee_{i \in M} \bigvee_{j \in M} \bigvee_{k \in H} \bigvee_{l \in H} (\mathit{Smokes}(i, \mathit{Chesterfields}) \wedge \mathit{Pet}(j, \mathit{Fox}) \wedge \mathit{LivesIn}(i, k) \wedge \mathit{LivesIn}(j, l) \wedge (|j - l| = 1)$

The Zebra Puzzle

Kools are smoked in the house next to the house where the horse is kept: $\bigvee_{i \in M} \bigvee_{j \in M} \bigvee_{k \in H} \bigvee_{l \in H} (\mathit{Smokes}(i, \mathit{Kools}) \wedge \mathit{Pet}(j, \mathit{Horse}) \wedge \mathit{LivesIn}(i, k) \wedge \mathit{LivesIn}(j, l) \wedge (|k - l| = 1))$

The Lucky Strike smoker drinks orange juice:
$\bigvee_{i \in M} (\mathit{Smokes}(i, \mathit{LuckyStrike}) \wedge \mathit{Drinks}(i, \mathit{OrangeJuice}))$

The Japanese smokes Parliaments: $\mathit{Smokes}(\mathit{Jap}, \mathit{Parliaments})$

The Norwegian lives next to the blue house:
$\bigvee_{i \in H} \bigvee_{j \in H} (\mathit{LivesIn}(\mathit{Nor}, i) \wedge (|i - j| = 1) \wedge \mathit{Color}(j, \mathit{Blue}))$

The Zebra Puzzle

The background assumptions:

$\text{exactly1}(\{\mathit{LivesIn}(m,h)|h \in H\})$ for each $m \in M$
$\text{exactly1}(\{\mathit{LivesIn}(m,h)|m \in M\})$ for each $h \in H$
$\text{exactly1}(\{\mathit{Smokes}(m,c)|c \in C\})$ for each $m \in M$
$\text{exactly1}(\{\mathit{Smokes}(m,c)|m \in M\})$ for each $c \in C$
$\text{exactly1}(\{\mathit{Pet}(m,a)|a \in A\})$ for each $m \in M$
$\text{exactly1}(\{\mathit{Pet}(m,a)|m \in M\})$ for each $a \in A$
$\text{exactly1}(\{\mathit{Color}(h,p)|p \in P\})$ for each $h \in H$
$\text{exactly1}(\{\mathit{Color}(h,p)|h \in H\})$ for each $p \in P$
$\text{exactly1}(\{\mathit{Drinks}(m,d)|m \in M\})$ for each $d \in D$
$\text{exactly1}(\{\mathit{Drinks}(m,d)|d \in D\})$ for each $m \in M$

Solving the puzzle:

- Feed all formulas to a SAT solver and run it (takes milliseconds)
- The satisfying valuation gives the solution

Predicate Logic: Motivation

Many application involve relational data

- objects
- sets of objects
- relations between objects

Propositional logic (Booleans only!) does not *directly* cover these

Predicate logic is the most basic logic of relations:

- Generalizes propositional logic
- Introduces features needed in complex applications
  - Concepts arising in data and knowledge management
  - Concepts arising in natural language processing
- Satisfiability & deduction have high computational complexity

Predicate Logic: Motivation

Many application involve relational data

- objects
- sets of objects
- relations between objects

Propositional logic (Booleans only!) does not *directly* cover these

Predicate logic is the most basic logic of relations:

- Generalizes propositional logic
- Introduces features needed in complex applications
  - Concepts arising in data and knowledge management
  - Concepts arising in natural language processing
- Satisfiability & deduction have high computational complexity

Comparison

Propositional Logic

- Valuations: $\begin{matrix} ABCDE \\ 01100 \end{matrix}$
- Formulas represent Boolean functions $\{0,1\}^n \to \{0,1\}$

Predicate Logic

- Universe $U = \{0, 1, 2, \ldots\}$ (finite or infinite set)
- Relations $R_i \subseteq U^{n_i}$
- Functions $f_j : U^{m_j} \to U$
- Example:
  - $U = \{0, 1, 2, 3, 4\}$
  - $P = \{(0, 1), (1, 2), (2, 3), (3, 4)\}$
  - $Q = \{0, 2, 4\}$
  - $f = \{(0, 1), (1, 2), (2, 0), (3, 3), (4, 0)\}$
- Formulas represent Boolean functions over relational facts

Comparison

Propositional Logic

- Valuations: $\begin{matrix} ABCDE \\ 01100 \end{matrix}$
- Formulas represent Boolean functions $\{0,1\}^n \to \{0,1\}$

Predicate Logic

- Universe $U = \{0, 1, 2, \ldots\}$ (finite or infinite set)
- Relations $R_i \subseteq U^{n_i}$
- Functions $f_j : U^{m_j} \to U$
- Example:
  - $U = \{0, 1, 2, 3, 4\}$
  - $P = \{(0, 1), (1, 2), (2, 3), (3, 4)\}$
  - $Q = \{0, 2, 4\}$
  - $f = \{(0, 1), (1, 2), (2, 0), (3, 3), (4, 0)\}$
- Formulas represent Boolean functions over relational facts

Predicate Logic: Example

Consider structure $\mathcal{S}$:

- $U = \{0, 1, 2, 3, 4\}$
- $a = 0$
  $b = 2$
  $c = 3$
- $P = \{(0, 1), (1, 2), (2, 3), (3, 4)\}$
- $Q = \{1, 3\}$
- $R = \{(0, 0, 0), (0, 1, 1), (1, 0, 1),$
  $(1, 1, 2), (0, 2, 2), (2, 0, 2)\}$

Following formulas are true in $\mathcal{S}$

1. $P(b, c)$
2. $\neg P(a, b)$
3. $\exists x.Q(x)$
4. $\exists x.R(a, b, x)$
5. $\exists x.(P(a, x) \wedge P(x, b) \wedge Q(x))$
6. $\forall x.(Q(x) \rightarrow \exists y.P(x, y))$

Predicate Logic: Example

Consider structure $\mathcal{S}$:

- $U = \{0, 1, 2, 3, 4\}$
- $a = 0$
  $b = 2$
  $c = 3$
- $P = \{(0, 1), (1, 2), (2, 3), (3, 4)\}$
- $Q = \{1, 3\}$
- $R = \{(0, 0, 0), (0, 1, 1), (1, 0, 1),$
  $(1, 1, 2), (0, 2, 2), (2, 0, 2)\}$

Following formulas are true in $\mathcal{S}$

1. $P(b, c)$
2. $\neg P(a, b)$
3. $\exists x.Q(x)$
4. $\exists x.R(a, b, x)$
5. $\exists x.(P(a, x) \wedge P(x, b) \wedge Q(x))$
6. $\forall x.(Q(x) \rightarrow \exists y.P(x, y))$

Symbols for Referring to a Structure

- Constant symbols c denote objects in the universe
- Function symbols f denote functions $U^n \to U$
  ($n \geq 1$ is the arity: how many arguments the function takes)
- Predicate symbols P denote relations $\subseteq U^n$
  (the arity $n \geq 1$ tells how many "columns")

Example

To talk about natural numbers $U = \{0, 1, 2, \ldots\}$ we could have

- constant symbol Zero
- function symbol Succ
- predicate GreaterThan

Symbols for Referring to a Structure

- Constant symbols c denote objects in the universe
- Function symbols f denote functions $U^n \to U$
  ($n \geq 1$ is the arity: how many arguments the function takes)
- Predicate symbols P denote relations $\subseteq U^n$
  (the arity $n \geq 1$ tells how many "columns")

Example

To talk about natural numbers $U = \{0, 1, 2, \ldots\}$ we could have

- constant symbol Zero
- function symbol Succ
- predicate GreaterThan

Syntax of Predicate Logic

1. Constant symbols and variables are terms
2. If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.
3. If P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an atomic formula. Also $t_1 = t_2$ is an atomic formula.
4. Atomic formulas are formulas
5. If ϕ ja ψ are formulas and x is a variable, then $(\neg\phi)$,$(\phi \vee \psi)$, $(\phi \wedge \psi)$,$(\phi \rightarrow \psi)$,$(\phi \leftrightarrow \psi)$,$(\forall x.\phi)$ and $(\exists x.\phi)$ are formulas.
6. Nothing else is a term, an atomic formula, or a formula.

Syntax of Predicate Logic

1. Constant symbols and variables are terms
2. If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.
3. If P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an atomic formula. Also $t_1 = t_2$ is an atomic formula.
4. Atomic formulas are formulas
5. If ϕ ja ψ are formulas and x is a variable, then $(\neg\phi)$,$(\phi \vee \psi)$, $(\phi \wedge \psi)$,$(\phi \rightarrow \psi)$,$(\phi \leftrightarrow \psi)$,$(\forall x.\phi)$ and $(\exists x.\phi)$ are formulas.
6. Nothing else is a term, an atomic formula, or a formula.

Syntax of Predicate Logic

1. Constant symbols and variables are terms
2. If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.
3. If P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an atomic formula. Also $t_1 = t_2$ is an atomic formula.
4. Atomic formulas are formulas
5. If ϕ ja ψ are formulas and x is a variable, then $(\neg\phi)$,$(\phi \vee \psi)$, $(\phi \wedge \psi)$,$(\phi \rightarrow \psi)$,$(\phi \leftrightarrow \psi)$,$(\forall x.\phi)$ and $(\exists x.\phi)$ are formulas.
6. Nothing else is a term, an atomic formula, or a formula.

Syntax of Predicate Logic

1. Constant symbols and variables are terms
2. If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.
3. If P is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an atomic formula. Also $t_1 = t_2$ is an atomic formula.
4. Atomic formulas are formulas
5. If ϕ ja ψ are formulas and x is a variable, then $(\neg\phi)$,$(\phi \vee \psi)$, $(\phi \wedge \psi)$,$(\phi \rightarrow \psi)$,$(\phi \leftrightarrow \psi)$,$(\forall x.\phi)$ and $(\exists x.\phi)$ are formulas.
6. Nothing else is a term, an atomic formula, or a formula.

Quantifiers $\forall$ and $\exists$

$\forall x.\Phi$ denotes "for all objects x the formula Φ holds"

$\exists x.\Phi$ denotes "for some object x the formula Φ holds"

($\forall x.\Phi$ is equivalent to $\neg\exists x.\neg\Phi$, and $\exists x.\Phi$ is equivalent to $\neg\forall x.\neg\Phi$!)

Examples:

- "All students are academics" $\forall x(Student(x) \rightarrow Academic(x))$
- "Some academics are Computer Scientists" $\exists x(Academic(x) \wedge CScientist(x))$ (Note: $\wedge$ vs $\rightarrow$ above)
- "No Computer Scientist is Dumb" $\forall x(CScientist(x) \rightarrow \neg Dumb(x))$
- "Everybody's got a friend" $\forall x \exists y(Friend(y, x))$ (Different friends?)
- "Somebody is everybody's boss" $\exists x \forall y(Boss(x, y))$ (The same boss)

Quantifiers $\forall$ and $\exists$

$\forall x.\Phi$ denotes "for all objects x the formula Φ holds"

$\exists x.\Phi$ denotes "for some object x the formula Φ holds"

($\forall x.\Phi$ is equivalent to $\neg\exists x.\neg\Phi$, and $\exists x.\Phi$ is equivalent to $\neg\forall x.\neg\Phi$!)

Examples:

- "All students are academics" $\forall x(Student(x) \rightarrow Academic(x))$
- "Some academics are Computer Scientists" $\exists x(Academic(x) \wedge CScientist(x))$ (Note: $\wedge$ vs $\rightarrow$ above)
- "No Computer Scientist is Dumb" $\forall x(CScientist(x) \rightarrow \neg Dumb(x))$
- "Everybody's got a friend" $\forall x \exists y(Friend(y, x))$ (Different friends?)
- "Somebody is everybody's boss" $\exists x \forall y(Boss(x, y))$ (The same boss)

Scopes of Quantifiers

- The scope of $\forall x$ in formula $\forall x.\phi$ is ϕ.
- The variable x is bound in its scope.
- A variable occurrence that is not bound is free.

Example

Consider variables in the following formula.

$$\forall \overbrace{x}^{\text{bound}} \left(P(\overbrace{x}^{\text{bound}}, \overbrace{y}^{\text{free}}, \overbrace{z}^{\text{free}}) \vee \exists \overbrace{y}^{\text{bound}} (Q(\overbrace{y}^{\text{bound}}) \rightarrow R(\overbrace{x}^{\text{bound}}, \overbrace{z}^{\text{free}})) \right).$$

Only those formulas can be assigned a truth value that only have bound variables.

Scopes of Quantifiers

- The scope of $\forall x$ in formula $\forall x.\phi$ is ϕ.
- The variable x is bound in its scope.
- A variable occurrence that is not bound is free.

Example

Consider variables in the following formula.

$$\forall \overbrace{x}^{\text{bound}} \left(P(\overbrace{x}^{\text{bound}}, \overbrace{y}^{\text{free}}, \overbrace{z}^{\text{free}}) \vee \exists \overbrace{y}^{\text{bound}} (Q(\overbrace{y}^{\text{bound}}) \rightarrow R(\overbrace{x}^{\text{bound}}, \overbrace{z}^{\text{free}})) \right).$$

Only those formulas can be assigned a truth value that only have bound variables.

Scopes of Quantifiers

- The scope of $\forall x$ in formula $\forall x.\phi$ is ϕ.
- The variable x is bound in its scope.
- A variable occurrence that is not bound is free.

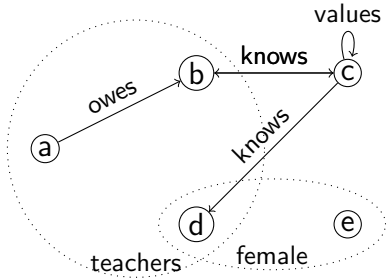
Example

Consider variables in the following formula.

$$\forall \overbrace{x}^{\text{bound}} \left(P(\overbrace{x}^{\text{bound}}, \overbrace{y}^{\text{free}}, \overbrace{z}^{\text{free}}) \vee \exists \overbrace{y}^{\text{bound}} (Q(\overbrace{y}^{\text{bound}}) \rightarrow R(\overbrace{x}^{\text{bound}}, \overbrace{z}^{\text{free}})) \right).$$

Only those formulas can be assigned a truth value that only have bound variables.

Example



$\exists x \exists y (Knows(x, y) \wedge Knows(y, x))$
$\exists x \exists y (Teacher(x) \wedge Teacher(y) \wedge Owes(x, y))$
$\exists x (V(x, x) \wedge \exists x (Teacher(x) \wedge Female(x)))$
$\neg \forall x (Teacher(x) \rightarrow Female(x))$
$\forall x (Knows(x, c) \rightarrow Owes(a, x))$

Example: The Peano Axioms for Arithmetics

1. $\forall x(Zero \neq Succ(x))$
2. $\forall x \forall y(Succ(x) = Succ(y) \rightarrow x = y)$
3. $\forall x(Plus(x, Zero) = x)$
4. $\forall x \forall y(Plus(x, Succ(y)) = Succ(Plus(x, y)))$
5. $\forall x(Mult(x, Zero) = Zero)$
6. $\forall x \forall y(Mult(x, Succ(y)) = Plus(Mult(x, y), x))$

Peanos's induction axiom is not (finitely) expressible in the predicate logic:

$$\begin{aligned}\forall y_1 \forall y_2 \cdots \forall y_k (&\psi(Zero, y_1, y_2, \ldots, y_k) \\ &\wedge \forall x(\psi(x, y_1, y_2, \ldots, y_k) \rightarrow \psi(Succ(x), y_1, y_2, \ldots, y_k)) \\ &\rightarrow \forall x \psi(x, y_1, y_2, \ldots, y_k)\end{aligned}$$

...for all formulas ψ!

Structures Defined Formally

Definition

A structure $\mathcal{S}$ (for *given constant, function and predicate symbols*) consists of the universe $U \neq \emptyset$ and interpretations $[\![...]\!]^{\mathcal{S}}$ of those symbols:

1. Constant symbols c are mapped to $[\![c]\!]^{\mathcal{S}} \in U$.
2. Variable symbols v are mapped to $[\![v]\!]^{\mathcal{S}} \in U$.
3. Function symbols f of arity n map to functions $[\![f]\!]^{\mathcal{S}} : U^n \to U$.
4. Predicate symbols P of arity n map to relations $[\![P]\!]^{\mathcal{S}} \subseteq U^n$.

Evaluation of Terms and Atomic Formulas

Interpretation of all symbols leads to interpretation $[\![t]\!]^{\mathcal{S}}$ of all terms:

- Constant symbols c are directly evaluated by $[\![c]\!]^{\mathcal{S}}$.
- Variables v are directly evaluated by $[\![v]\!]^{\mathcal{S}}$.
- For terms $f(t_1, \ldots, t_n)$: $[\![f(t_1, \ldots, t_n)]\!]^{\mathcal{S}} = [\![f]\!]^{\mathcal{S}}([\![t_1]\!]^{\mathcal{S}}, \ldots, [\![t_n]\!]^{\mathcal{S}})$

For atomic formulas $P(t_1, \ldots, t_n)$ we define truth in $\mathcal{S}$:

$$\mathcal{S} \models P(t_1, \ldots, t_n) \text{ iff } ([\![t_1]\!]^{\mathcal{S}}, \ldots, [\![t_n]\!]^{\mathcal{S}}) \in [\![P]\!]^{\mathcal{S}}$$
$$\mathcal{S} \models (t_1 = t_2) \text{ iff } [\![t_1]\!]^{\mathcal{S}} = [\![t_2]\!]^{\mathcal{S}}$$

Evaluation of Terms and Atomic Formulas

Interpretation of all symbols leads to interpretation $[\![t]\!]^{\mathcal{S}}$ of all terms:

- Constant symbols c are directly evaluated by $[\![c]\!]^{\mathcal{S}}$.
- Variables v are directly evaluated by $[\![v]\!]^{\mathcal{S}}$.
- For terms $f(t_1, \ldots, t_n)$: $[\![f(t_1, \ldots, t_n)]\!]^{\mathcal{S}} = [\![f]\!]^{\mathcal{S}}([\![t_1]\!]^{\mathcal{S}}, \ldots, [\![t_n]\!]^{\mathcal{S}})$

For atomic formulas $P(t_1, \ldots, t_n)$ we define truth in $\mathcal{S}$:

$$\begin{aligned} \mathcal{S} \models P(t_1, \ldots, t_n) \quad &\text{iff} \quad ([\![t_1]\!]^{\mathcal{S}}, \ldots, [\![t_n]\!]^{\mathcal{S}}) \in [\![P]\!]^{\mathcal{S}} \\ \mathcal{S} \models (t_1 = t_2) \quad &\text{iff} \quad [\![t_1]\!]^{\mathcal{S}} = [\![t_2]\!]^{\mathcal{S}} \end{aligned}$$

Evaluation of Formulas

Definition

Let $\mathcal{S}$ be a structure and U its universe. For all formulas α and β and variables v we define:

1. For atomic formulas α, $\mathcal{S} \models \alpha$ as on the previous slide
2. Define $\mathcal{S} \models \alpha \wedge \beta$, $\mathcal{S} \models \alpha \vee \beta$ and $\mathcal{S} \models \neg\alpha$ as in propositional logic
3. $\mathcal{S} \models \exists v.\alpha$ iff $\mathcal{S}[v \mapsto a] \models \alpha$ for some $a \in U$
4. $\mathcal{S} \models \forall v.\alpha$ iff $\mathcal{S}[v \mapsto a] \models \alpha$ for all $a \in U$

Here $\mathcal{S}[v \mapsto a]$ is a structure exactly like $\mathcal{S}$ except that it maps v to a.

Reasoning Tasks in Predicate Logic

As in propositional logic:

- ϕ is satisfiable iff $\mathcal{S} \models \phi$ for some $\mathcal{S}$
- ϕ is valid iff for $\mathcal{S} \models \phi$ for all $\mathcal{S}$
- $\psi \models \phi$ iff $\mathcal{S} \models \phi$ for all $\mathcal{S}$ s.t. $\mathcal{S} \models \psi$

These all (implicitly) involve going through all (infinitely many) possible structures (with the given constant, function and predicate symbols): this is the source of the very high complexity of these tasks.
(Methods for testing satisfiability and logical consequence are not exhaustively going through (finite and infinite) structures, but do something more clever.)

Knowledge Representation: Open World Assumption

Everything that is not explicitly stated nor inferable from explicitly stated, is unknown

Example

John is male. Mary is female. Jack is male.
John is sick. Mary is sick.

Is Jack sick? We don't know.

This is inherent in (almost) all logic, but often forgotten. Pay attention!

Knowledge Representation: Definitions

Non-recursive definitions are straightforward.

Example

$\forall x(\text{grandMother}(x) \leftrightarrow (\text{female}(x) \wedge \exists y \exists z.(\text{parent}(x, y) \wedge \text{parent}(y, z))))$

Knowledge Representation: Definitions

Example

Network with (directed) links L and connections C (link sequences):

$L(a, b) \wedge L(a, c) \wedge L(b, d) \wedge L(d, e)$
$\forall x.C(x, x)$
$\forall x \forall y \forall z(C(x, y) \wedge C(y, z) \rightarrow C(x, z))$
$\forall x \forall y(L(x, y) \rightarrow C(x, y))$

Is there a connection from e to c? We don't know.

Note: The above formulas are perfectly OK as long as we don't expect to infer *lack* of connections or links.

Knowledge Representation: Definitions

Example (Version 2)

$\forall x \forall y.(L(x,y) \leftrightarrow ((x=a \wedge y=b) \vee (x=a \wedge y=c) \vee (x=b \wedge y=d) \vee (x=d \wedge y=e))$
$\forall x \forall y(C(x,y) \leftrightarrow (L(x,y) \vee (x=y) \vee \exists z(C(x,z) \wedge C(z,y))))$

Problem *not* solved: $\mathcal{S}$ such that $[\![L]\!]^{\mathcal{S}} = \{(a,b),(a,c),(b,d),(d,e)\}$ and $[\![C]\!]^{\mathcal{S}} = \{a,b,c,d,e\}^2$ satisfies the above formulas.

Issue: $\{a,b,c,d,e\}^2$ isn't a set-inclusion minimal interpretation of C

Predicate logic cannot express transitive closure (Fagin 1974)

Knowledge Representation: Definitions

Example (Version 2)

$\forall x \forall y.(L(x,y) \leftrightarrow ((x=a \wedge y=b) \vee (x=a \wedge y=c) \vee (x=b \wedge y=d) \vee (x=d \wedge y=e))$
$\forall x \forall y(C(x,y) \leftrightarrow (L(x,y) \vee (x=y) \vee \exists z(C(x,z) \wedge C(z,y))))$

Problem *not* solved: $\mathcal{S}$ such that $[\![L]\!]^{\mathcal{S}} = \{(a,b),(a,c),(b,d),(d,e)\}$ and $[\![C]\!]^{\mathcal{S}} = \{a,b,c,d,e\}^2$ satisfies the above formulas.

Issue: $\{a,b,c,d,e\}^2$ isn't a set-inclusion minimal interpretation of C

Predicate logic cannot express transitive closure (Fagin 1974)

Knowledge Representation: Definitions

Example (Version 2)

$\forall x \forall y.(L(x,y) \leftrightarrow ((x=a \wedge y=b) \vee (x=a \wedge y=c) \vee (x=b \wedge y=d) \vee (x=d \wedge y=e))$
$\forall x \forall y(C(x,y) \leftrightarrow (L(x,y) \vee (x=y) \vee \exists z(C(x,z) \wedge C(z,y))))$

Problem *not* solved: $\mathcal{S}$ such that $[\![L]\!]^{\mathcal{S}} = \{(a,b),(a,c),(b,d),(d,e)\}$ and $[\![C]\!]^{\mathcal{S}} = \{a,b,c,d,e\}^2$ satisfies the above formulas.

Issue: $\{a,b,c,d,e\}^2$ isn't a set-inclusion minimal interpretation of C

Predicate logic cannot express transitive closure (Fagin 1974)

Knowledge Representation: Definitions

Bounded-length connections can be expressed:

Example (Version 3)

$\forall x \forall y.(L(x,y) \leftrightarrow ((x=a \wedge y=b) \vee (x=a \wedge y=c) \vee (x=b \wedge y=d) \vee (x=d \wedge y=e))$
$\forall x(C_0(x,y) \leftrightarrow (x=y))$
$\forall x \forall y(C_1(x,y) \leftrightarrow (L(x,y) \vee x=y))$
$\forall x \forall y(C_i(x,y) \leftrightarrow (C_i(x,y) \vee \exists z(L(x,z) \wedge C_{i-1}(z,y))))$, for $i \in \{1,2,\ldots,n\}$

Here the cycle in the definition of C has been broken.
This is a good solution when *it is known* that there are $\leq n$ nodes.

General solution: Use fixpoint logics instead

Application: Semantic Web

- Current web search is based on textual match with search terms
- Any hits with "animals that use sonar but are not either bats or dolphins"?
- OWL and other description logics (DL) attempt to fix this problem:
  - Web documents and data are annotated with their meaning
  - Search with complex logic-based queries
- DL are based on Predicate Logic (+ modal logics)
  - Satisfiability and $\models$ for Predicate Logic in general undecidable
  - Much research to identify DL fragments with a lower complexity
  - Several poly-time fragments of DL known

Application: Semantic Web

- Current web search is based on textual match with search terms
- Any hits with "animals that use sonar but are not either bats or dolphins"?
- OWL and other description logics (DL) attempt to fix this problem:
  - Web documents and data are annotated with their meaning
  - Search with complex logic-based queries
- DL are based on Predicate Logic (+ modal logics)
  - Satisfiability and $\models$ for Predicate Logic in general undecidable
  - Much research to identify DL fragments with a lower complexity
  - Several poly-time fragments of DL known

Application: Semantic Web

- Current web search is based on textual match with search terms
- Any hits with "animals that use sonar but are not either bats or dolphins"?
- OWL and other description logics (DL) attempt to fix this problem:
  - Web documents and data are annotated with their meaning
  - Search with complex logic-based queries
- DL are based on Predicate Logic (+ modal logics)
  - Satisfiability and $\models$ for Predicate Logic in general undecidable
  - Much research to identify DL fragments with a lower complexity
  - Several poly-time fragments of DL known

Description Logics

DL syntax	example	predicate logic
C	Student	$\text{Student}(x)$
$C_1 \sqcap \cdots \sqcap C_n$	Student⊓CScientist	$\text{Student}(x) \wedge \text{CScientist}(x)$
$C_1 \sqcup \cdots \sqcup C_n$	Student⊔Graduate	$\text{Student}(x) \vee \text{Graduate}(x)$
$\neg C$	¬Student	$\neg\text{Student}(x)$
$\forall R.C$	∀sibling.Female	$\forall y(\text{sibling}(x, y) \wedge \text{Female}(y))$
$\exists R.C$	∃child.Student	$\exists y(\text{child}(x, y) \wedge \text{Student}(y))$

Note: DL expressions implicitly have one free variable

Note: DL syntax (typically) excludes n-ary predicates for $n \geq 3$

Description Logics

DL syntax	example	predicate logic
C	Student	$\text{Student}(x)$
$C_1 \sqcap \cdots \sqcap C_n$	Student⊓CScientist	$\text{Student}(x) \wedge \text{CScientist}(x)$
$C_1 \sqcup \cdots \sqcup C_n$	Student⊔Graduate	$\text{Student}(x) \vee \text{Graduate}(x)$
$\neg C$	¬Student	$\neg\text{Student}(x)$
$\forall R.C$	∀sibling.Female	$\forall y(\text{sibling}(x, y) \wedge \text{Female}(y))$
$\exists R.C$	∃child.Student	$\exists y(\text{child}(x, y) \wedge \text{Student}(y))$

Note: DL expressions implicitly have one free variable

Note: DL syntax (typically) excludes n-ary predicates for $n \geq 3$

Description Logics

syntax	example	predicate logic
$C_1 \sqsubseteq C_2$	Student$\sqsubseteq$Human	$\forall x(\text{Student}(x) \rightarrow \text{Human}(x))$
$C_1 \equiv C_2$	Theria$\equiv$Marsupialia$\sqcup$Placentalia	$\forall x(\text{Theria}(x) \leftrightarrow \text{Marsupialia}(x) \vee \text{Placentalia}(x))$

In DL knowledge bases, one usually does the following split:

TBox – Terminological box
Definitions of concepts (classes):

Mammalia$\equiv$Theria$\sqcup$Monotremata
Theria$\equiv$Marsupialia$\sqcup$Placentalia
Monotremata$\equiv$Platypoda$\sqcup$Tachyglossa
...
Ornithorhyncus$\equiv$Platypus

ABox – Assertional box
Facts about individual objects:

Bruce : Platypus, Bruce : Male,
Sheila : WesternLongBeakedEchidna,
Sheila : Female,
(Bruce,Sheila) : friend

Logic and Relational Algebra

- Natural join as $\forall x \forall y \forall z \forall u (P(x, y, z) \wedge Q(y, z, u) \leftrightarrow R(x, y, z, u))$
- Projection $\forall x \forall y \forall z (P(x, y, z) \leftrightarrow R(x, y))$
- Selection $\forall x \forall y \forall z (P(x, y, z) \wedge Q(x, y, z) \leftrightarrow R(x, y, z))$
- Union $\forall x \forall y \forall z (P(x, y, z) \vee Q(x, y, z) \leftrightarrow R(x, y, z))$
- Intersection $\forall x \forall y \forall z (P(x, y, z) \wedge Q(x, y, z) \leftrightarrow R(x, y, z))$
- Difference $\forall x \forall y \forall z (P(x, y, z) \wedge \neg Q(x, y, z) \leftrightarrow R(x, y, z))$

Remarks:

- The R relation is the result of the operation on the lhs of $\leftrightarrow$)
- For union, intersection and difference the columns don't need to be the same

Logic and Databases

Datalog is a language for deductive databases, offering inference mechanisms not present in e.g. SQL.

Example

Datalog

```
mother(X,Y) :- parent(X,Y), female(X).
grandmother(X,Y) :- mother(X,Z),
                    parent(Z,Y).
```

Predicate logic

$$\forall x \forall y (\text{parent}(x,y) \wedge \text{female}(x) \rightarrow \text{mother}(x,y))$$
$$\forall x \forall y \forall z (\text{mother}(x,z) \wedge \text{parent}(z,y) \rightarrow \text{grandmother}(x,y))$$

Implementations of Datalog exist, but are not in very wide use.
Features and implementation technology developed for Datalog have been incorporated in commercial products like IBM DB2 and the SQL:1999 standard

Reduction to Propositional Logic?

High complexity of Predicate Logic makes it less attractive:

- Logical consequence and satisfiability might not be needed:
  - Evaluate formulas w.r.t. a given structure (e.g. RDBMS, graph, ...)
- If they are needed:
  1. Universe fixed, finite ("small")? Reduce to Propositional Logic (next slide)!
  2. Otherwise, use Predicate Logic theorem provers (scalability?)

Reduction to Propositional Logic

Assumptions:

- No function symbols
- Domain Closure Assumption: All objects (in universe) have names
  For constant symbols $\{a, b, c\}$ this corresponds to
  $\forall x.((x = a) \vee (x = b) \vee (x = c))$
- Unique Names Assumption: No object has two or more names
  For constant symbols $\{a, b, c\}$ this corresponds to
  $\neg(a = b) \wedge \neg(a = c) \wedge \neg(b = c)$

Reduction to Propositional Logic by eliminating $\exists$, $\forall$:

- $\forall x \Phi(x) \equiv \bigwedge_{c \in C} \Phi(c/x)$
- $\exists x \Phi(x) \equiv \bigvee_{c \in C} \Phi(c/x)$

Semantics of Natural Language

- Very difficult topic
- Lots of theories, some technically very deep and sophisticated
- Some theories work well with narrow domains and vocabularies
- None work well with general unrestricted language

Next: Montague grammar or Montague semantics

Higher-Order Functions and Types

Standard definition of functions: mapping between set-theoretic objects, e.g. $\mathbb{N}^n \to \mathbb{N}$ which expressible as a set $\subseteq \mathbb{N}^{n+1}$

Higher-order functions are mappings from functions, or to functions

Two examples:

- Map $n : \mathbb{N}$ to function $f : \mathbb{N} \to \mathbb{N}$ such that $f(x) = x + n$
  The type of this mapping is $\mathbb{N} \to (\mathbb{N} \to \mathbb{N})$
- Map function $f : \mathbb{N} \to \mathbb{N}$ to function $g : \mathbb{N} \to \mathbb{N}$ such that $g(x) = f(x) + 1$ for all $x \in NAT$
  The type of this mapping is $(\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to \mathbb{N})$

Types can be arbitrarily complex, e.g. $(\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to (\mathbb{N} \to \mathbb{N}))$

Higher-Order Functions and Types

Standard definition of functions: mapping between set-theoretic objects, e.g. $\mathbb{N}^n \to \mathbb{N}$ which expressible as a set $\subseteq \mathbb{N}^{n+1}$

Higher-order functions are mappings from functions, or to functions

Two examples:

- Map $n : \mathbb{N}$ to function $f : \mathbb{N} \to \mathbb{N}$ such that $f(x) = x + n$
  The type of this mapping is $\mathbb{N} \to (\mathbb{N} \to \mathbb{N})$
- Map function $f : \mathbb{N} \to \mathbb{N}$ to function $g : \mathbb{N} \to \mathbb{N}$ such that $g(x) = f(x) + 1$ for all $x \in NAT$
  The type of this mapping is $(\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to \mathbb{N})$

Types can be arbitrarily complex, e.g. $(\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to (\mathbb{N} \to \mathbb{N}))$

The λ Notation

Definition

$\lambda x.f(x)$ denotes an unnamed function that expresses the mapping f.
$(\lambda x.f(x))\ c$ means applying $\lambda x.f(x)$ to c, with result $f(c)$.

Example

- $\lambda x.(x+1)$ (integer incrementation)
  - $(\lambda x.(x+1))\ 5 = 5{+}1 = 6$
- $\lambda x.\lambda y.(x+y)$ ("Curried" function for integer addition)
  - $(\lambda x.\lambda y.(x+y))\ 5 = \lambda y.(5+y)$
  - $(\lambda x.\lambda y.(x+y))\ 5\ 7 = (\lambda y.(5+y))\ 7 = 5+7 = 12$

Montague Grammar

- Natural language semantics developed by Richard Montague (published in 1970-1973)
  Related work: categorial grammar
- Grammar non-terminals associated with higher-order functions
  - Verb phrase (VP): $term \rightarrow formula$
  - Noun phrase (NP): $(term \rightarrow formula) \rightarrow formula$
  - Common noun (CN): $term \rightarrow formula$
  - Determiners (DET): $(term \rightarrow formula) \rightarrow ((term \rightarrow formula) \rightarrow formula)$
  - Transitive verb (TV): $term \rightarrow (term \rightarrow formula)$
- Sentences (S) are associated with *formulas*, with e.g. parse NP VP obtained by function application NP(VP)

Montague Grammar

- Natural language semantics developed by Richard Montague (published in 1970-1973)
  Related work: categorial grammar
- Grammar non-terminals associated with higher-order functions
  - Verb phrase (VP): $\mathit{term} \to \mathit{formula}$
  - Noun phrase (NP): $(\mathit{term} \to \mathit{formula}) \to \mathit{formula}$
  - Common noun (CN): $\mathit{term} \to \mathit{formula}$
  - Determiners (DET): $(\mathit{term} \to \mathit{formula}) \to ((\mathit{term} \to \mathit{formula}) \to \mathit{formula})$
  - Transitive verb (TV): $\mathit{term} \to (\mathit{term} \to \mathit{formula})$
- Sentences (S) are associated with *formulas*, with e.g. parse NP VP obtained by function application NP(VP)

Small Grammar with Semantics

S	:	$NP\ VP$	$(NP\ VP)$
NP	:	$name$	$\lambda P.(P\ name)$
	|	$DET\ CN$	$(DET\ CN)$
	|	$DET\ RCN$	$(DET\ RCN)$
DET	:	"$some$"	$\lambda P.\lambda Q.\exists x((P\ x)\wedge(Q\ x))$
	|	"a"	$\lambda P.\lambda Q.\exists x((P\ x)\wedge(Q\ x))$
	|	"$every$"	$\lambda P.\lambda Q.\forall x((P\ x)\rightarrow(Q\ x))$
	|	"no"	$\lambda P.\lambda Q.\forall x((P\ x)\rightarrow(\neg(Q\ x)))$
VP	:	$intransverb$	$\lambda x.intransverb(x)$
	|	$TV\ NP$	$\lambda x.(NP\ (\lambda y.(TV\ y\ x)))$
TV	:	$transverb$	$\lambda x.\lambda y.transverb(x,y)$
RCN	:	CN "$that$" VP	$\lambda x.((CN\ x)\wedge(VP\ x))$
	|	CN "$that$" $NP\ TV$	$\lambda x.((CN\ x)\wedge(NP\ (\lambda y.(TV\ y\ x))))$
CN	:	$predicate$	$\lambda x.predicate(x)$

(All variables x introduced in the DET rules must be "new".)

Discussion

Why are the types of NP and VP *not*

- Noun phrase (NP): *term*
- Verb phrase (VP): *term* $\rightarrow$ *formula*

and application (VP NP), but application is (NP VP) and the types are as follows?

- Noun phrase (NP): (*term* $\rightarrow$ *formula*) $\rightarrow$ *formula*
- Verb phrase (VP): *term* $\rightarrow$ *formula*

In predicate logic a term refers to an individual, so why is this different?

While a term will indeed refer to an individual, the term will often need to be qualified in complex ways. Consider "The man that sings runs":

- Intransitive verb "runs" corresponds to $\lambda x.runs(x)$
- But having $x =''$ *man*$''$ is not enough, as we also need "that sings"

Discussion

Why are the types of NP and VP *not*

- Noun phrase (NP): *term*
- Verb phrase (VP): $term \rightarrow formula$

and application (VP NP), but application is (NP VP) and the types are as follows?

- Noun phrase (NP): $(term \rightarrow formula) \rightarrow formula$
- Verb phrase (VP): $term \rightarrow formula$

In predicate logic a term refers to an individual, so why is this different?

While a term will indeed refer to an individual, the term will often need to be qualified in complex ways. Consider "The man that sings runs":

- Intransitive verb "runs" corresponds to $\lambda x.runs(x)$
- But having $x =''$ *man*$''$ is not enough, as we also need "that sings"

Example

expression	meaning
a	$\lambda P.\lambda Q.\exists x((P\ x) \wedge (Q\ x))$
man	$\lambda x.MAN(x)$
a man	$\lambda Q.\exists x(MAN(x) \wedge (Q\ x))$
sleeps	$\lambda x.SLEEPS(x)$
a man sleeps	$\exists x(MAN(x) \wedge SLEEPS(x))$
man that sweats	$\lambda x.(MAN(x) \wedge SWEATS(x))$
a man that sweats	$\lambda Q.\exists x(MAN(x) \wedge SWEATS(x) \wedge (Q\ x))$
a man that sweats sleeps	$\exists x(MAN(x) \wedge SWEATS(x) \wedge SLEEPS(x))$

Example

English sentence	translation to logic
Sirpa sees Jarmo	sees(sirpa,jarmo)
every woman sees a man	$\forall x(woman(x) \rightarrow (\exists y(man(y) \wedge sees(x,y))))$
every woman sees a man that sleeps	$\forall x(woman(x) \rightarrow (\exists y(man(y) \wedge sleeps(y) \wedge sees(x,y))))$
a woman that eats sees a man that sleeps	$\exists x(woman(x) \wedge eats(x) \wedge \exists y(man(y) \wedge sleeps(y) \wedge sees(x,y)))$

Can be further reduced to SQL for querying RDBMS!

What We Do (and Do Not) Have

Method for reducing complex fragments of natural language to logic.

- Useful in domains with a limited vocabulary
  - Small number of concepts, meaning of concepts well-defined
  - Answering queries e.g. from a database

Unlimited language e.g. books, newspapers, speech very difficult:

- Few concepts connect to other concepts *logically*
  - Rare exceptions: definitions like "bachelor" $\sim$ "unmarried adult male"
- Homonyms (disambiguation of the sense of the word)
- Meaning of many concepts is fuzzy
- Many phenomena in pragmatics not covered by logic
  - (Conversational) implicature
  - Presupposition

Fuzziness of Concepts

- Definition: "Chair is something the *purpose* of which is to be sat on (by one person)"
- What about a broken chair you cannot sit on?
- You could use a non-chair object as a chair! Is it a chair then?
- Chairs: objects similar to usual chairs or otherwise suitable for sitting ("family resemblance")

Conclusion:

- There is no precise definition of chair
- Why would you even care about a definition of chair?
- But the word "chair" still provides useful information. What is it?

Fuzziness of Concepts

- Definition: "Chair is something the *purpose* of which is to be sat on (by one person)"
- What about a broken chair you cannot sit on?
- You could use a non-chair object as a chair! Is it a chair then?
- Chairs: objects similar to usual chairs or otherwise suitable for sitting ("family resemblance")

Conclusion:

- There is no precise definition of chair
- Why would you even care about a definition of chair?
- But the word "chair" still provides useful information. What is it?

Implicature

In human communication, what is explicitly said, often is not the actual message.

1. "I wonder what time it is." (message: "Please tell me the time.")
2. "I just heard the bells toll 8 times." (message: "It's a bit past 8.")

Often something is implied (suggested):

- "Some students didn't go." (implied: "Some students did go.")
- "I haven't finished homework yet." (implied: "I already started the homework.")
- "They have a son." (implied: "The have only one son and no daughters.")

Depending on the context, the implied claim would not necessarily have to hold.

Implicature

In human communication, what is explicitly said, often is not the actual message.

1. "I wonder what time it is." (message: "Please tell me the time.")
2. "I just heard the bells toll 8 times." (message: "It's a bit past 8.")

Often something is implied (suggested):

- "Some students didn't go." (implied: "Some students did go.")
- "I haven't finished homework yet." (implied: "I already started the homework.")
- "They have a son." (implied: "The have only one son and no daughters.")

Depending on the context, the implied claim would not necessarily have to hold.

Presupposition

What is said explicitly, implicitly assumes other facts

- "My mom thinks that my wife is clever" presupposes "I have a wife"
- "I regret not trying to quit smoking" presupposes
  - "I am (still) smoking"
  - "I didn't try to quit smoking"

Assumptions often defeasible (something may cancel them):

- "John did not pass the exam."
- "Actually, John didn't even go to the exam!"

Pragmatics, NLP and A.I.

Lots of deep work in linguistics on pragmatics, but...

- these theories are too descriptive from an A.I. viewpoint,
- not systematic and detailed enough to implement.

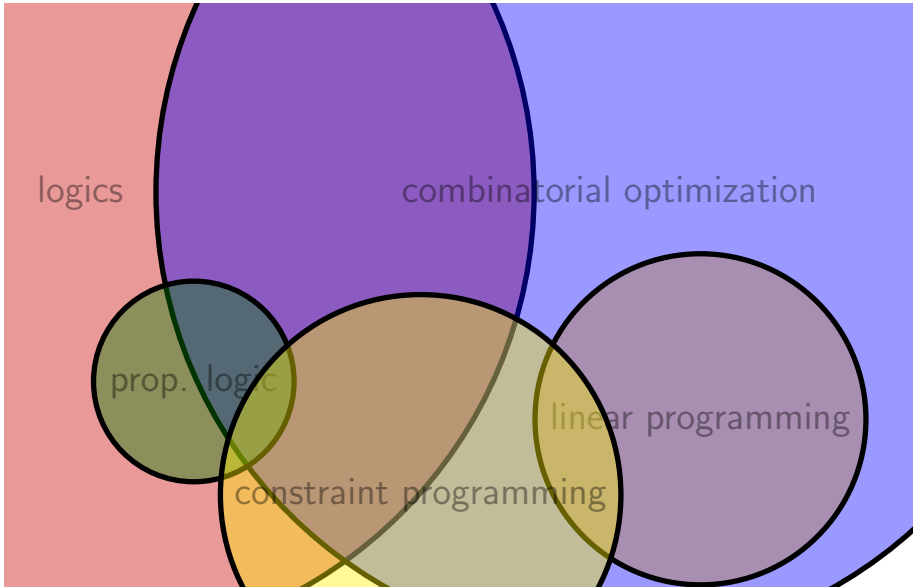
Not at all clear where and how human-level A.I. in terms of natural language is going to arise!!!

Successful NLP Applications

Fundamental NLP problems remain unsolved, hence no deep NLP:

- IBM Watson (Jeopardy): clever lookup from text databases, no real understanding
- Siri, Alexa, chatbots, ...: ad hoc matching of sentences to pre-defined syntactic patterns
- Machine translation: usable, but no expert-level translation in sight

Logic in the Big Picture



Logic in the Big Picture

- Boolean constraints (SAT, Integer Programming, Constraint Programming)
- Arithmetic constraints (SAT+numbers, Integer/Linear Programming, Constraint Programming)
- Relational constraints (Predicate Logic, Constraint Programming)

Examples:

- Mixed Integer-Linear Programming: integers+reals
- Boolean satisfiability (SAT): Boolean
- SMT (SAT modulo Theories): Booleans+integers+reals+bitvectors+others