Quick Course Overview

- Quick review of logic and computational problems
- What the course is about
- First assignment

First-order Logic

$\forall$ **x Student(x)** $\rightarrow$ **Honest(x)**

Every student is honest

$\exists$ **x Student(x)** $\wedge$ **Honest(x)**

Some student is honest

$\forall$ **x Student(x)** $\rightarrow$ $\exists$ **y Professor(y)** $\wedge$ **Advises(y,x)**

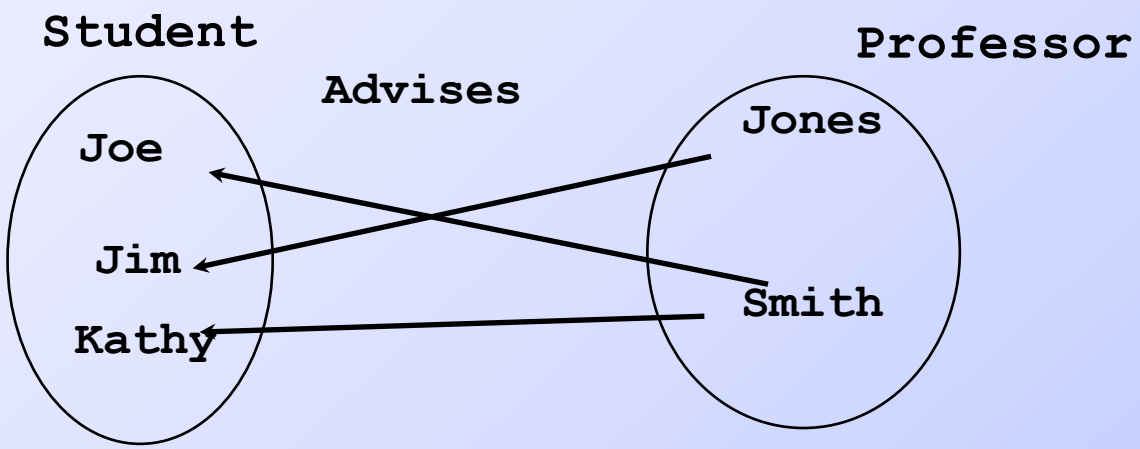
Every student is advised by a professor

Semantics of FO Logic

`Student(x), Professor(y),`
`Advises(x,y)`

In FO logic we have a **signature** (aka **vocabulary**): set of predicates, functions, constants

Each signature has many possible **interpretations**

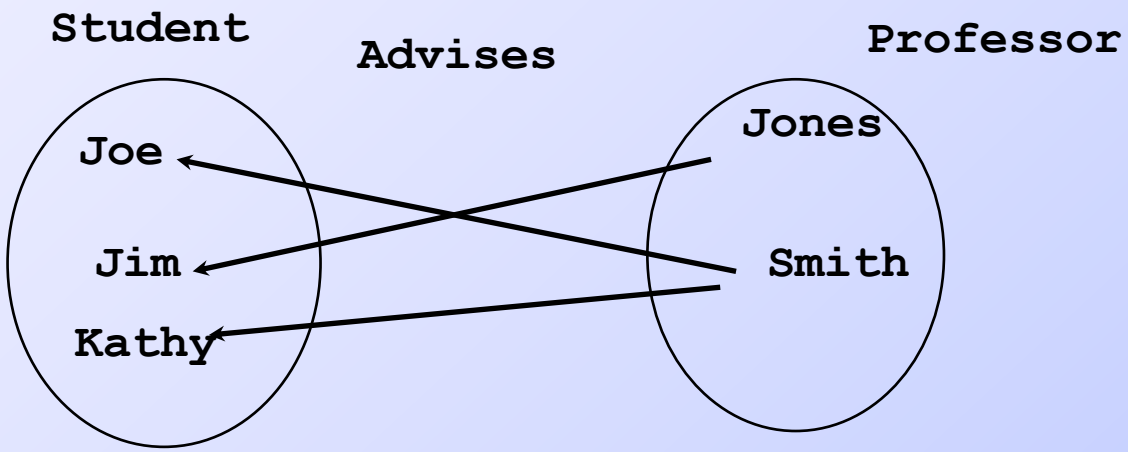


Semantics of FO: By Example

$\forall x\,[\mathrm{Student}(x) \rightarrow \exists y\ (\mathrm{Advises}(y,x) \wedge \mathrm{Professor}(y))]$

$\exists x\ (\mathrm{Student}(x) \wedge \mathrm{Professor}(x))$

$\exists x\ \exists y\ \mathrm{Student}(x) \wedge \mathrm{Student}(y) \wedge \neg(x=y)$
$\wedge\ \exists z\ (\mathrm{Advises}(z,x) \wedge \mathrm{Advises}(z,y))$



Computational Problems for FO

A sentence ϕ is **satisfiable** if there is **some** interpretation $\mathfrak{M}$ where ϕ is true

A sentence ϕ is **valid** (or, is a **tautology**) if ϕ is true in **every** interpretation

A sentence ϕ is a **contradiction** if it is **false in every interpretation**

Two sentences ϕ_1 and ϕ_2 are **equivalent** if they are true in exactly the same interpretations

Examples

$\forall x \ (A(x) \vee \neg A(x))$ Valid

$A(c) \rightarrow \exists y \, A(y)$ Valid

$\forall x \, (Man(x) \rightarrow Mortal(x))$

$\forall z \ z+z>z$

History

The question of decidability of satisfiability and validity of first order logic was introduced by the mathematician David Hilbert in the 1920's, with the goal of using logic as a way of automatically verifying mathematics.

History

Kurt Godel created the first complete proof system for first order logic.

Using proofs, he gave a **semi-decision procedure** (one-way test) for validity, equivalence, implication of first-order logic sentences.

History

Alonzo Church and Alan Turing showed that the decision problem for first-order logic is **undecidable.**

- Cannot decide if ϕ is satisifiable
- Cannot decide if ϕ is valid
- Cannot decide if ϕ_1 implies ϕ_2

Why is FO satisfiability undecidable?

We want to know if there is a structure that satisfies ϕ

- Infinitely many structures to check
- ϕ may be satisfied, but only in a structure with a very complex shape

What is to be Done?

Satisfiability and Validity are undecidable.

Response 1) Restrict **structures**:
there are classes of interpretations, where the satisfiability problem **relative to that set of structures** is decidable.

Response 2) Restrict to **special first-order sentences**:
Find fragments of FO (restricted kinds of sentences) for which satisfiability and/or validity are decidable.

Restricting Shape of Models

E.g. **Vocabulary={Advises(x,y)}**

Restrict structures to be linear orders.

There are algorithms that will take a first-order sentence ϕ using just **<** and decide if it holds in all linear orders, and similarly whether it holds on some linear order.

Joe —**Advises**→ **Susan** —**Advises**→ **David** —**Advises**→ **Marcelo**

Restricting Models

E.g. **Vocabulary={`Advises(x,y)`, $a_1(x) \ldots a_n(x)$}**

Restrict to (finite) **words (labeled orderings)** :
< is required to be a linear order, labels $a_1 \ldots a_n$ are restricted to be disjoint, one of them must be true for each element of the order.

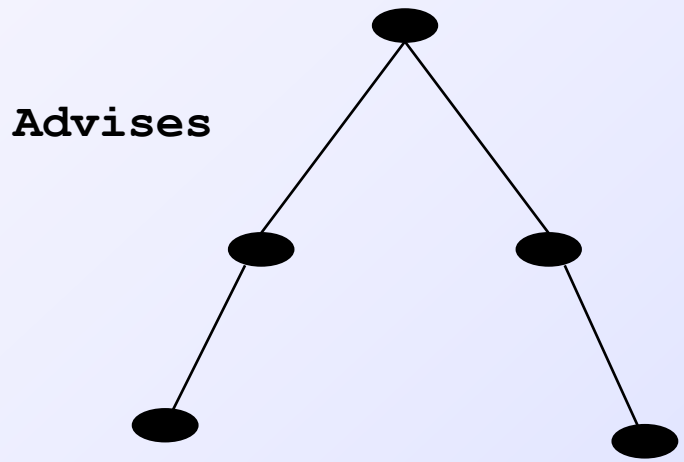
There are algorithms that will take a first-order sentence ϕ using just **<** and $a_1 \ldots a_n$ decide if it holds in all labelled linear orders.

These are based on converting the sentence to a word automaton and deciding if the language of the automaton is non-empty.

Restricting Models

E.g. Vocabulary={`Advises(x,y)`, $a_1(x) \ldots a_n(x)$}

Restrict to **finite labeled trees** : `Advises` required to be a tree, $a_1 \ldots a_n$ are restricted to be disjoint, one of them must be true for each element of the domain.

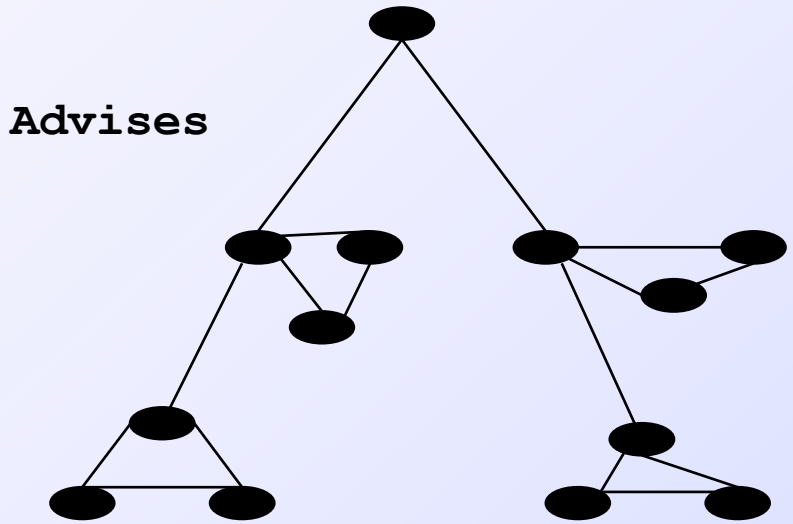


These are based on converting the sentence to a tree automaton and deciding if the language of the automaton is non-empty.

Restricting Models

E.g. **Vocabulary={`Advises(x,y)`, $a_1(x) \ldots a_n(x)$: $\Sigma=\{a_1 \ldots a_n\}$}**

Restrict to finite labeled **tree-like graphs** : `Advises` required to be a graph of some fixed **tree-width** while $a_1 \ldots a_n$ are restricted as before.



These are based on converting the sentence to a tree automaton running over tree codes of the structures and deciding if the language of the automaton is non-empty.

What is to be Done?

Satisfiability and Validity are undecidable.

Response 2: Restrict sentences.
Find fragments of first-order logic (restricted kinds of sentences) for which satisfiability and/or validity are decidable.

Many examples:

- Modal Logic
- Guarded Fragment
- Guarded Negation Fragment
- Unary Negation Fragment
- ...

Example: Guarded Fragment

1995 Guarded Fragment (Andreka, Nemeti, Van Bentham)

Start with basic relations (as in general FO), and close under

- boolean operations ($\wedge$, $\neg$, $\vee$) as before
- guarded quantifiers

$\forall\, \mathbf{x_1 \ldots x_n}\ \mathbf{R(x_1 \ldots x_n)} \rightarrow \phi\mathbf{(x_1 \ldots x_n)}$

$\exists\, \mathbf{x_1 \ldots x_n}\ \mathbf{R(x_1 \ldots x_n)} \wedge \phi\mathbf{(x_1 \ldots x_n)}$

Guarded Fragment

$\forall$ **x Artist(x)** $\rightarrow$ **Person(x)**

$\forall$ **x y Man(x)** $\wedge$ **Marriedto(x,y)** $\leftrightarrow$ **Husband(x)**

Can be converted to GF

Decidability of GF

One can get an algorithm that decides whether
a GF formula is satisfiable, valid, etc.

Main idea behind decidability: GF sentences cannot force a structure to be complicated.

If a GF sentence ϕ over graphs is satisfied, it is satisfied in a graph which is tree-like (has tree-width bounded by a number computable from ϕ)

Now use previous results on “special structures”.

This course

Go through examples of restricted logics that are decidable by this technique.

- **Tree model property:** Show that any satisfiable sentence has a tree-like model.
- **Translation**: Show that sentence ϕ in the logic can be translated effectively into a sentence ϕ^* such that:
  evaluating ϕ^* on the tree code of a tree-like model is equivalent to evaluating ϕ over the model.
- Decision procedures over trees: decide whether ϕ^* is satisfiable over trees that encode models using decision procedures for logics over trees (goes via converting ϕ^* to an automaton and checking non-emptiness).

Variation: directly translate ϕ into an automaton checking ϕ^* with the property above.

First case study: Modal Logic

Chapter 4 of the course notes or
Section 5 of Vardi, *Why is Modal Logic so Robustly Decidable?*

- Read the definition of Modal Logic

- Prove Proposition 5.2 (for every model there is a tree that is “bisimilar to it”) and Proposition 5.1 (if two structures are bisimilar, they agree on all modal formulas – Vardi shows this for a more general logic with fixpoints).

Together these show that every satisfiable modal formula has a tree model.

- Argue (as Vardi does not) that the tree can be made finite.

- Show how to construct from a modal logic formula a tree automaton that accepts finite tree models of the formula.

Structure