ECE750T-28: Computer-aided Reasoning for Software Engineering

Lecture 7: Validity Proofs and Properties of FOL

Vijay Ganesh
(Original notes from Isil Dillig)

Overview

- Last lecture: Started talking about formal semantics for FOL

Overview

- Last lecture: Started talking about formal semantics for FOL
- Agenda for today:

Overview

- Last lecture: Started talking about formal semantics for FOL
- Agenda for today:
  - Finish semantics of FOL

Overview

- Last lecture: Started talking about formal semantics for FOL
- Agenda for today:
  - Finish semantics of FOL
  - Semantics argument method for proving FOL validity

Overview

- Last lecture: Started talking about formal semantics for FOL
- Agenda for today:
  - Finish semantics of FOL
  - Semantics argument method for proving FOL validity
  - Important properties of FOL

Review

- We evaluate formulas F under structure $S = \langle U, I \rangle$ and variable assignment σ.

Review

- We evaluate formulas F under structure $S = \langle U, I \rangle$ and variable assignment σ.
- If F evaluates to true under U, I, σ, we write $U, I, \sigma \models F$

Review

- We evaluate formulas F under structure $S = \langle U, I \rangle$ and variable assignment σ.
- If F evaluates to true under U, I, σ, we write $U, I, \sigma \models F$
- If F evaluates to false under U, I, σ, we write $U, I, \sigma \not\models F$

Review

- We evaluate formulas F under structure $S = \langle U, I \rangle$ and variable assignment σ.
- If F evaluates to true under U, I, σ, we write $U, I, \sigma \models F$
- If F evaluates to false under U, I, σ, we write $U, I, \sigma \not\models F$
- Semantics of $\models$ defined inductively.

Review

- We evaluate formulas F under structure $S = \langle U, I \rangle$ and variable assignment σ.
- If F evaluates to true under U, I, σ, we write $U, I, \sigma \models F$
- If F evaluates to false under U, I, σ, we write $U, I, \sigma \not\models F$
- Semantics of $\models$ defined inductively.
- Already defined semantics of terms, predicates, and logical connectives

Example

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

Example

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$p(f(b), f(x)) \rightarrow p(f(x), f(b) \qquad =$$

Example

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$p(f(b), f(x)) \rightarrow p(f(x), f(b) \quad = \quad \mathit{true}$$

Example

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{rcl} p(f(b), f(x)) \rightarrow p(f(x), f(b) & = & \mathit{true} \\ p(f(x), f(b)) \rightarrow p(f(b), f(x)) & = & \end{array}$$

Example

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{lcl} p(f(b), f(x)) \rightarrow p(f(x), f(b) & = & \mathit{true} \\ p(f(x), f(b)) \rightarrow p(f(b), f(x)) & = & \mathit{false} \end{array}$$

Variant of Variable Assignment

- We still need to evaluate formulas containing quantifiers!

Variant of Variable Assignment

- We still need to evaluate formulas containing quantifiers!
- But to do that, we first define an x-variant of a variable assignment.

Variant of Variable Assignment

- We still need to evaluate formulas containing quantifiers!
- But to do that, we first define an x-variant of a variable assignment.
- An x-variant of assignment σ, written $\sigma[x \mapsto c]$, is the assignment that agrees with σ for assignments to all variables except x and assigns x to c.

Variant of Variable Assignment

- We still need to evaluate formulas containing quantifiers!
- But to do that, we first define an x-variant of a variable assignment.
- An x-variant of assignment σ, written $\sigma[x \mapsto c]$, is the assignment that agrees with σ for assignments to all variables except x and assigns x to c.
- Example: If $\sigma : \{x \mapsto 1, y \mapsto 2\}$, what is $\sigma[x \mapsto 3]$?

Variant of Variable Assignment

- We still need to evaluate formulas containing quantifiers!
- But to do that, we first define an x-variant of a variable assignment.
- An x-variant of assignment σ, written $\sigma[x \mapsto c]$, is the assignment that agrees with σ for assignments to all variables except x and assigns x to c.
- Example: If $\sigma : \{x \mapsto 1, y \mapsto 2\}$, what is $\sigma[x \mapsto 3]$? $\sigma : \{x \mapsto 3, y \mapsto 2\}$

Evaluation of Formulas II

- We can now give semantics to quantifiers:

Evaluation of Formulas II

- We can now give semantics to quantifiers:
- Universal quantifier:

$$U, I, \sigma \models \forall x.F \quad \text{iff} \quad \text{for all } o \in U,\ U, I, \sigma[x \mapsto o] \models F$$

Evaluation of Formulas II

- We can now give semantics to quantifiers:
- Universal quantifier:

$$U, I, \sigma \models \forall x.F \quad \text{iff} \quad \text{for all } o \in U,\ U, I, \sigma[x \mapsto o] \models F$$

- Existential quantifier:

$$U, I, \sigma \models \exists x.F \quad \text{iff} \quad \text{there exists } o \in U \text{ s.t } U, I, \sigma[x \mapsto o] \models F$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\forall x.p(x, a) \qquad\qquad =$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\forall x. p(x, a) \qquad = \quad \mathit{false}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$
\begin{gathered}
I(a) = \star \quad I(b) = \bullet \\
I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\} \\
I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}
\end{gathered}
$$

- Under U, I and σ, what do these formulas evaluate to?

$$
\begin{array}{lcl}
\forall x. p(x, a) & = & \mathit{false} \\
\forall x. p(b, x) & = &
\end{array}
$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{lcl} \forall x.p(x,a) & = & \mathit{false} \\ \forall x.p(b,x) & = & \mathit{true} \end{array}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$\begin{array}{c} I(a) = \star \quad I(b) = \bullet \\ I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\} \\ I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\} \end{array}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{lcl} \forall x.p(x, a) & = & \mathit{false} \\ \forall x.p(b, x) & = & \mathit{true} \\ \exists x.p(a, x) & = & \end{array}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{lcl} \forall x. p(x, a) & = & \mathit{false} \\ \forall x. p(b, x) & = & \mathit{true} \\ \exists x. p(a, x) & = & \mathit{false} \end{array}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$\begin{array}{c} I(a) = \star \quad I(b) = \bullet \\ I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\} \\ I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\} \end{array}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{rcl} \forall x. p(x, a) & = & \mathit{false} \\ \forall x. p(b, x) & = & \mathit{true} \\ \exists x. p(a, x) & = & \mathit{false} \\ \forall x. \ (p(a, x) \rightarrow p(b, x)) & = & \end{array}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$
\begin{array}{c}
I(a) = \star \quad I(b) = \bullet \\
I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\} \\
I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}
\end{array}
$$

- Under U, I and σ, what do these formulas evaluate to?

$$
\begin{array}{ccc}
\forall x. p(x, a) & = & \mathit{false} \\
\forall x. p(b, x) & = & \mathit{true} \\
\exists x. p(a, x) & = & \mathit{false} \\
\forall x. \ (p(a, x) \rightarrow p(b, x)) & = & \mathit{true}
\end{array}
$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{rcl} \forall x. p(x, a) & = & \mathit{false} \\ \forall x. p(b, x) & = & \mathit{true} \\ \exists x. p(a, x) & = & \mathit{false} \\ \forall x. \ (p(a, x) \rightarrow p(b, x)) & = & \mathit{true} \\ \exists x. \ (p(f(x), f(x)) \rightarrow p(x, x)) & = & \end{array}$$

Example III: Evaluation of Formulas

- Consider universe $\{\star, \bullet\}$, variable assignment $\sigma : \{x \mapsto \star\}$, and interpretation I:

$$I(a) = \star \quad I(b) = \bullet$$
$$I(f) = \{\star \mapsto \bullet, \bullet \mapsto \star\}$$
$$I(p) = \{\langle \bullet, \star \rangle, \langle \bullet, \bullet \rangle\}$$

- Under U, I and σ, what do these formulas evaluate to?

$$\begin{array}{lcl} \forall x. p(x, a) & = & \mathit{false} \\ \forall x. p(b, x) & = & \mathit{true} \\ \exists x. p(a, x) & = & \mathit{false} \\ \forall x. \ (p(a, x) \rightarrow p(b, x)) & = & \mathit{true} \\ \exists x. \ (p(f(x), f(x)) \rightarrow p(x, x)) & = & \color{red}{\mathit{true}} \end{array}$$

Satisfiability and Validity of First-Order Formulas

- A first-order formula F is satisfiable iff there exists a structure S and variable assignment σ such that

$$S, \sigma \models F$$

Satisfiability and Validity of First-Order Formulas

- A first-order formula F is satisfiable iff there exists a structure S and variable assignment σ such that

$$S, \sigma \models F$$

- Otherwise, F is unsatisfiable.

Satisfiability and Validity of First-Order Formulas

- A first-order formula F is satisfiable iff there exists a structure S and variable assignment σ such that

$$S, \sigma \models F$$

- Otherwise, F is unsatisfiable.
- A structure S is a model of F, written $S \models F$, if for all variable assignments σ, $S, \sigma \models F$.

Satisfiability and Validity of First-Order Formulas

- A first-order formula F is satisfiable iff there exists a structure S and variable assignment σ such that

$$S, \sigma \models F$$

- Otherwise, F is unsatisfiable.

- A structure S is a model of F, written $S \models F$, if for all variable assignments σ, $S, \sigma \models F$.

- A first-order formula F is valid, written $\models F$ if for all structures S, $S, \sigma \models F$

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable?

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation:

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle\star, \star\rangle\}$

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle \star, \star \rangle\}$
- Is this formula valid?

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle\star, \star\rangle\}$
- Is this formula valid? no

Satisfiability and Validity Examples

- Is the formula $\forall x. \exists y. p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle \star, \star \rangle\}$
- Is this formula valid? no
- Falsifying interpretation:

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle\star, \star\rangle\}$
- Is this formula valid? no
- Falsifying interpretation: $U = \{\star\}$, $I(p) = \{\}$

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle\star, \star\rangle\}$
- Is this formula valid? no
- Falsifying interpretation: $U = \{\star\}$, $I(p) = \{\}$
- Is the formula $\forall x.(p(x, x) \rightarrow \exists y.p(x, y))$ valid?

Satisfiability and Validity Examples

- Is the formula $\forall x. \exists y. p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle \star, \star \rangle\}$
- Is this formula valid? no
- Falsifying interpretation: $U = \{\star\}$, $I(p) = \{\}$
- Is the formula $\forall x. (p(x, x) \rightarrow \exists y. p(x, y))$ valid? yes

Satisfiability and Validity Examples

- Is the formula $\forall x.\exists y.p(x, y)$ satisfiable? yes
- Satisfying interpretation: $U = \{\star\}$, $I(p) = \{\langle\star, \star\rangle\}$
- Is this formula valid? no
- Falsifying interpretation: $U = \{\star\}$, $I(p) = \{\}$
- Is the formula $\forall x.(p(x, x) \rightarrow \exists y.p(x, y))$ valid? yes
- Intuition: Consider any object o. If $p(o, o)$ is false, then implication satisfied. If $p(o, o)$ is true, there there exists a y (namely o) s.t $p(x, y)$ is also true.

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid?

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ:

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation:

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid?

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid

More Satisfiability and Validity Examples

- Is the formula $(\exists x. p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x. p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x. (p(x) \rightarrow q(x))) \rightarrow (\exists x. (p(x) \wedge q(x)))$?

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle \star \rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x.(p(x) \rightarrow q(x))) \rightarrow (\exists x.(p(x) \land q(x)))$? contingent

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x.(p(x) \rightarrow q(x))) \rightarrow (\exists x.(p(x) \wedge q(x)))$? contingent
- Satisfying interpretation:

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x.(p(x) \rightarrow q(x))) \rightarrow (\exists x.(p(x) \wedge q(x)))$? contingent
- Satisfying interpretation: $U = \{\star\}, I(p) = \{\langle\star\rangle\}, I(q) = \{\langle\star\rangle\}$

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x.(p(x) \rightarrow q(x))) \rightarrow (\exists x.(p(x) \wedge q(x)))$? contingent
- Satisfying interpretation: $U = \{\star\}, I(p) = \{\langle\star\rangle\}, I(q) = \{\langle\star\rangle\}$
- Falsifying interpretation:

More Satisfiability and Validity Examples

- Is the formula $(\exists x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? contingent
- Satisfying U, I, σ: $U = \{\star, \circ\}, I(p) = \{\}, \sigma(x) = \circ$
- Falsifying interpretation: $U = \{\star, \circ\}, I(p) = \{\langle\star\rangle\}, \sigma(x) = \circ$
- Is the formula $(\forall x.p(x)) \rightarrow p(x)$ contingent, unsat, or valid? valid
- What about $(\forall x.(p(x) \rightarrow q(x))) \rightarrow (\exists x.(p(x) \land q(x)))$? contingent
- Satisfying interpretation: $U = \{\star\}, I(p) = \{\langle\star\rangle\}, I(q) = \{\langle\star\rangle\}$
- Falsifying interpretation: $U = \{\star\}, I(p) = \{\}, I(q) = \{\langle\star\rangle\}$

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F?

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F? not necessarily

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F? not necessarily
- Consider a sentence F such that $S, \sigma \models F$. Is S a model F?

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F? not necessarily
- Consider a sentence F such that $S, \sigma \models F$. Is S a model F? yes

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F? not necessarily
- Consider a sentence F such that $S, \sigma \models F$. Is S a model F? yes
- Consider a ground formula F such that $S, \sigma \models F$. Is S a model of F?

True/False Exercises

- Consider a formula F such that $S, \sigma \models F$. Is S a model F? not necessarily
- Consider a sentence F such that $S, \sigma \models F$. Is S a model F? yes
- Consider a ground formula F such that $S, \sigma \models F$. Is S a model of F? yes

Motivation for semantic argument method

- So far, we defined what it means for a first-order formula to be satisfiable and valid.

Motivation for semantic argument method

- So far, we defined what it means for a first-order formula to be satisfiable and valid.
- However, we haven't talked about how to prove that a formula in FOL is valid.

Motivation for semantic argument method

- So far, we defined what it means for a first-order formula to be satisfiable and valid.
- However, we haven't talked about how to prove that a formula in FOL is valid.
- Will use semantic argument method to prove validity of first-order formulas

Motivation for semantic argument method

- So far, we defined what it means for a first-order formula to be satisfiable and valid.
- However, we haven't talked about how to prove that a formula in FOL is valid.
- Will use semantic argument method to prove validity of first-order formulas
- Extension of same technique from propositional logic

Duality of Satisfiability and Validity

- Recall: In propositional logic, satisfiability and validity are dual concepts:

$$\boxed{F \text{ is valid iff } \neg F \text{ is unsatisfiable}}$$

Duality of Satisfiability and Validity

- Recall: In propositional logic, satisfiability and validity are dual concepts:

$$\boxed{F \text{ is valid iff } \neg F \text{ is unsatisfiable}}$$

- This duality also holds in first-order logic.

Duality of Satisfiability and Validity

- Recall: In propositional logic, satisfiability and validity are dual concepts:

$$\boxed{F \text{ is valid iff } \neg F \text{ is unsatisfiable}}$$

- This duality also holds in first-order logic.
- Thus, if we have a technqiue for deciding validity in FOL, this immediately yields a way to decide satisfiability.

Duality of Satisfiability and Validity

- Recall: In propositional logic, satisfiability and validity are dual concepts:

  $$\boxed{F \text{ is valid iff } \neg F \text{ is unsatisfiable}}$$

- This duality also holds in first-order logic.
- Thus, if we have a technqiue for deciding validity in FOL, this immediately yields a way to decide satisfiability.
- Hence, we'll only focus on proving validity in this lecture

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.
- This technique is not particularly amenable to automation, but is useful for paper-and-pencil proofs of validity.

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.
- This technique is not particularly amenable to automation, but is useful for paper-and-pencil proofs of validity.
- Recall: Semantic argument method is a proof by contradiction.

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.
- This technique is not particularly amenable to automation, but is useful for paper-and-pencil proofs of validity.
- Recall: Semantic argument method is a proof by contradiction.
- Basic idea: Assume that F is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.
- This technique is not particularly amenable to automation, but is useful for paper-and-pencil proofs of validity.
- Recall: Semantic argument method is a proof by contradiction.
- Basic idea: Assume that F is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$
- Then, apply proof rules.

Semantic Argument Method to Prove Validity

- We will use the semantic argument technique from earlier to prove validity of first-order formulas.
- This technique is not particularly amenable to automation, but is useful for paper-and-pencil proofs of validity.
- Recall: Semantic argument method is a proof by contradiction.
- Basic idea: Assume that F is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$
- Then, apply proof rules.
- If can derive contradiction on every branch of proof, F is valid.

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S, \sigma \ \models \ \neg F}{S, \sigma \ \not\models \ F}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S, \sigma \ \models \ \neg F}{S, \sigma \ \not\models \ F} \qquad \frac{S, \sigma \ \not\models \ \neg F}{}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S, \sigma \ \models \ \neg F}{S, \sigma \ \not\models \ F} \qquad \frac{S, \sigma \ \not\models \ \neg F}{S, \sigma \ \models \ F}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S, \sigma \ \models \ \neg F}{S, \sigma \ \not\models \ F} \qquad \frac{S, \sigma \ \not\models \ \neg F}{S, \sigma \ \models \ F}$$

- And elimination rule:

$$\frac{S, \sigma \ \models \ F \wedge G}{}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S, \sigma \ \models \ \neg F}{S, \sigma \ \not\models \ F} \qquad \frac{S, \sigma \ \not\models \ \neg F}{S, \sigma \ \models \ F}$$

- And elimination rule:

$$\frac{S, \sigma \ \models \ F \land G}{\begin{array}{l} S, \sigma \ \models \ F \\ S, \sigma \ \models \ G \end{array}}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S,\sigma \ \models \ \neg F}{S,\sigma \ \not\models \ F} \qquad \frac{S,\sigma \ \not\models \ \neg F}{S,\sigma \ \models \ F}$$

- And elimination rule:

$$\frac{S,\sigma \ \models \ F \wedge G}{\begin{array}{l} S,\sigma \ \models \ F \\ S,\sigma \ \models \ G \end{array}} \qquad \frac{S,\sigma \ \not\models \ F \wedge G}{}$$

Proof Rules I (Review)

- All proof rules from prop. logic carry over to first-order logic.
- As before, proof rules come in pairs, for each connective, we have one case for $\models$, one case for $\not\models$
- Negation elimination:

$$\frac{S,\sigma \;\models\; \neg F}{S,\sigma \;\not\models\; F} \qquad \frac{S,\sigma \;\not\models\; \neg F}{S,\sigma \;\models\; F}$$

- And elimination rule:

$$\frac{S,\sigma \;\models\; F \wedge G}{\begin{array}{l} S,\sigma \;\models\; F \\ S,\sigma \;\models\; G \end{array}} \qquad \frac{S,\sigma \;\not\models\; F \wedge G}{S,\sigma \;\not\models\; F \quad \mid \quad S,\sigma \;\not\models\; G}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \models F \vee G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \models F \vee G}{S, \sigma \models F \quad | \quad S, \sigma \models G}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \ \models \ F \vee G}{S, \sigma \ \models \ F \ \mid \ S, \sigma \ \models \ G} \qquad \frac{S, \sigma \ \not\models \ F \vee G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \ \models \ F \lor G}{S,\sigma \ \models \ F \quad | \quad S,\sigma \ \models \ G} \qquad \frac{S,\sigma \ \not\models \ F \lor G}{\begin{array}{l} S,\sigma \ \not\models \ F \\ S,\sigma \ \not\models \ G \end{array}}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \ \models \ F \vee G}{S,\sigma \ \models \ F \quad | \quad S,\sigma \ \models \ G} \qquad \frac{S,\sigma \ \not\models \ F \vee G}{\begin{array}{l} S,\sigma \ \not\models \ F \\ S,\sigma \ \not\models \ G \end{array}}$$

- Implication elimination:

$$\frac{S,\sigma \ \models \ F \rightarrow G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \models F \vee G}{S, \sigma \models F \quad | \quad S, \sigma \models G} \qquad \frac{S, \sigma \not\models F \vee G}{\begin{array}{l} S, \sigma \not\models F \\ S, \sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S, \sigma \models F \rightarrow G}{S, \sigma \not\models F \quad | \quad S, \sigma \models G}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \models F \vee G}{S,\sigma \models F \quad | \quad S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \vee G}{\begin{array}{l} S,\sigma \not\models F \\ S,\sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S,\sigma \models F \rightarrow G}{S,\sigma \not\models F \quad | \quad S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \rightarrow G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \;\models\; F \vee G}{S,\sigma \;\models\; F \quad | \quad S,\sigma \;\models\; G} \qquad \frac{S,\sigma \;\not\models\; F \vee G}{\begin{array}{l} S,\sigma \;\not\models\; F \\ S,\sigma \;\not\models\; G \end{array}}$$

- Implication elimination:

$$\frac{S,\sigma \;\models\; F \rightarrow G}{S,\sigma \;\not\models\; F \quad | \quad S,\sigma \;\models\; G} \qquad \frac{S,\sigma \;\not\models\; F \rightarrow G}{\begin{array}{l} S,\sigma \;\models\; F \\ S,\sigma \;\not\models\; G \end{array}}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \models F \vee G}{S, \sigma \models F \quad | \quad S, \sigma \models G} \qquad \frac{S, \sigma \not\models F \vee G}{\begin{array}{l} S, \sigma \not\models F \\ S, \sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S, \sigma \models F \rightarrow G}{S, \sigma \not\models F \quad | \quad S, \sigma \models G} \qquad \frac{S, \sigma \not\models F \rightarrow G}{\begin{array}{l} S, \sigma \models F \\ S, \sigma \not\models G \end{array}}$$

- If and only if elimination:

$$\frac{S, \sigma \models F \leftrightarrow G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \models F \vee G}{S,\sigma \models F \quad | \quad S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \vee G}{\begin{array}{l} S,\sigma \not\models F \\ S,\sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S,\sigma \models F \rightarrow G}{S,\sigma \not\models F \quad | \quad S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \rightarrow G}{\begin{array}{l} S,\sigma \models F \\ S,\sigma \not\models G \end{array}}$$

- If and only if elimination:

$$\frac{S,\sigma \models F \leftrightarrow G}{S,\sigma \models F \wedge G \quad | \quad S,\sigma \models \neg F \wedge \neg G}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S,\sigma \models F \vee G}{S,\sigma \models F \;\mid\; S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \vee G}{\begin{array}{l} S,\sigma \not\models F \\ S,\sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S,\sigma \models F \rightarrow G}{S,\sigma \not\models F \;\mid\; S,\sigma \models G} \qquad \frac{S,\sigma \not\models F \rightarrow G}{\begin{array}{l} S,\sigma \models F \\ S,\sigma \not\models G \end{array}}$$

- If and only if elimination:

$$\frac{S,\sigma \models F \leftrightarrow G}{S,\sigma \models F \wedge G \;\mid\; S,\sigma \models \neg F \wedge \neg G}$$

$$\frac{S,\sigma \not\models F \leftrightarrow G}{}$$

Proof Rules II (Review)

- Or elimination:

$$\frac{S, \sigma \models F \vee G}{S, \sigma \models F \quad | \quad S, \sigma \models G} \qquad \frac{S, \sigma \not\models F \vee G}{\begin{array}{l} S, \sigma \not\models F \\ S, \sigma \not\models G \end{array}}$$

- Implication elimination:

$$\frac{S, \sigma \models F \rightarrow G}{S, \sigma \not\models F \quad | \quad S, \sigma \models G} \qquad \frac{S, \sigma \not\models F \rightarrow G}{\begin{array}{l} S, \sigma \models F \\ S, \sigma \not\models G \end{array}}$$

- If and only if elimination:

$$\frac{S, \sigma \models F \leftrightarrow G}{S, \sigma \models F \wedge G \quad | \quad S, \sigma \models \neg F \wedge \neg G}$$

$$\frac{S, \sigma \not\models F \leftrightarrow G}{S, \sigma \models F \wedge \neg G \quad | \quad S, \sigma \models \neg F \wedge G}$$

Proof Rules III (New)

- We need new rules to eliminate universal and existential quantifiers.

Proof Rules III (New)

- We need new rules to eliminate universal and existential quantifiers.
- Universal elimination I:

$$\frac{U, I, \sigma \models \forall x.F}{}$$

Proof Rules III (New)

- We need new rules to eliminate universal and existential quantifiers.
- Universal elimination I:

$$\frac{U, I, \sigma \;\models\; \forall x.F}{U, I, \sigma[x \mapsto o] \;\models\; F} \text{ (for any } o \in U)$$

Proof Rules III (New)

- We need new rules to eliminate universal and existential quantifiers.
- Universal elimination I:

$$\frac{U, I, \sigma \ \models \ \forall x.F}{U, I, \sigma[x \mapsto o] \ \models \ F} \ (\text{for } \textcolor{red}{\text{any}} \ \ o \in U)$$

- Example: Suppose $U, I, \sigma \models \forall x.hates(jack, x)$

Proof Rules III (New)

- We need new rules to eliminate universal and existential quantifiers.
- Universal elimination I:

$$\frac{U, I, \sigma \;\models\; \forall x.F}{U, I, \sigma[x \mapsto o] \;\models\; F} \text{ (for any } \; o \in U)$$

- Example: Suppose $U, I, \sigma \models \forall x.hates(jack, x)$
- Using the above proof rule, we can conclude:

$$U, I, \sigma[x \mapsto I(jack)] \models hates(jack, x)$$

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \ \not\models \ \forall x.F}{}$$

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \;\not\models\; \forall x.F}{U, I, \sigma[x \mapsto o] \;\not\models\; F}(\text{for a fresh } o \in U)$$

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \ \not\models \ \forall x.F}{U, I, \sigma[x \mapsto o] \ \not\models \ F} \text{(for a fresh } o \in U\text{)}$$

- By a fresh object constant, we mean an object that has not been previously used in the proof

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \ \not\models \ \forall x.F}{U, I, \sigma[x \mapsto o] \ \not\models \ F} \text{(for a fresh } o \in U\text{)}$$

- By a fresh object constant, we mean an object that has not been previously used in the proof

- Why do we have this restriction?

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \;\not\models\; \forall x.F}{U, I, \sigma[x \mapsto o] \;\not\models\; F} \text{(for a fresh } o \in U\text{)}$$

- By a fresh object constant, we mean an object that has not been previously used in the proof
- Why do we have this restriction?
- If U, I, σ do not entail $\forall x.F$, we know there is some object for which F does not hold, but we don't know which one

Universal Elimation Rule II

- Universal elimination II:

$$\frac{U, I, \sigma \;\not\models\; \forall x.F}{U, I, \sigma[x \mapsto o] \;\not\models\; F} (\text{for a fresh } o \in U)$$

- By a fresh object constant, we mean an object that has not been previously used in the proof
- Why do we have this restriction?
- If U, I, σ do not entail $\forall x.F$, we know there is some object for which F does not hold, but we don't know which one
- If we have have used an object o before in the proof, we might know something else about o

Existential Elimination Rule 1

- Existential elimination I:

$$\frac{U, I, \sigma \models \exists x.F}{}$$

Existential Elimination Rule 1

- Existential elimination I:

$$\frac{U, I, \sigma \ \models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \models \ F} (\text{for a fresh } o \in U)$$

Existential Elimination Rule 1

- Existential elimination I:

$$\frac{U, I, \sigma \ \models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \models \ F} (\text{for a fresh } o \in U)$$

- Again, fresh means an object that has not been used before

Existential Elimination Rule 1

- Existential elimination I:

$$\frac{U, I, \sigma \ \models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \models \ F} \text{(for a fresh } o \in U)$$

- Again, fresh means an object that has not been used before
- If U, I, σ entail $\exists x.F$, we know there is some object for which F holds, but we don't know which object

Existential Elimination Rule 1

- Existential elimination I:

$$\frac{U, I, \sigma \ \models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \models \ F} (\text{for a fresh } o \in U)$$

- Again, fresh means an object that has not been used before
- If U, I, σ entail $\exists x.F$, we know there is some object for which F holds, but we don't know which object
- If we introduce an object o we have previously used, we might know something else about o

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \not\models \exists x.F}{\quad}$$

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \ \not\models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \not\models \ F} (\text{for any } o \in U)$$

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \;\not\models\; \exists x.F}{U, I, \sigma[x \mapsto o] \;\not\models\; F} \text{(for any } o \in U)$$

- Why can we instantiate x with any object?

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \not\models \exists x.F}{U, I, \sigma[x \mapsto o] \not\models F} (\text{for any } o \in U)$$

- Why can we instantiate x with any object?
- Because if U, I, σ do not entail $\exists x.F$, this means there does not exist any object for which F holds

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \ \not\models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \not\models \ F} (\text{for any } o \in U)$$

- Why can we instantiate x with any object?
- Because if U, I, σ do not entail $\exists x.F$, this means there does not exist any object for which F holds
- Thus, no matter what object x maps to, it still won't entail F

Existential Elimination Rule II

- Existantial elimination II:

$$\frac{U, I, \sigma \ \not\models \ \exists x.F}{U, I, \sigma[x \mapsto o] \ \not\models \ F} (\text{for any } o \in U)$$

- Why can we instantiate x with any object?
- Because if U, I, σ do not entail $\exists x.F$, this means there does not exist any object for which F holds
- Thus, no matter what object x maps to, it still won't entail F
- Therefore, ok to instantiate x with any object, regardless of whether it has been used before

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons
- Contradiction rule:

$$\frac{\begin{array}{lcl} U, I, \sigma & \models & p(s_1, \ldots, s_n) \\ U, I, \sigma & \not\models & p(t_1, \ldots, t_n) \end{array}}{}$$

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons
- Contradiction rule:

$$\frac{\begin{array}{l} U, I, \sigma \models p(s_1, \ldots, s_n) \\ U, I, \sigma \not\models p(t_1, \ldots, t_n) \\ (I, \sigma)(s_i) = (I, \sigma)(t_i) \text{ for all } i \in [1, n] \end{array}}{}$$

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons
- Contradiction rule:

$$\frac{\begin{array}{l} U, I, \sigma \models p(s_1, \ldots, s_n) \\ U, I, \sigma \not\models p(t_1, \ldots, t_n) \\ (I, \sigma)(s_i) = (I, \sigma)(t_i) \text{ for all } i \in [1, n] \end{array}}{U, I, \sigma \models \bot}$$

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons
- Contradiction rule:

$$\frac{\begin{array}{l} U, I, \sigma \models p(s_1, \ldots, s_n) \\ U, I, \sigma \not\models p(t_1, \ldots, t_n) \\ (I, \sigma)(s_i) = (I, \sigma)(t_i) \text{ for all } i \in [1, n] \end{array}}{U, I, \sigma \models \bot}$$

- Example: Suppose we have $S, \{x \mapsto a\} \models p(x)$ and $S, \{y \mapsto a\} \not\models p(y)$

Proof Rules V (New)

- Finally, we need a rule for deriving for contradicitons
- Contradiction rule:

$$\frac{\begin{array}{l} U, I, \sigma \models p(s_1, \ldots, s_n) \\ U, I, \sigma \not\models p(t_1, \ldots, t_n) \\ (I, \sigma)(s_i) = (I, \sigma)(t_i) \text{ for all } i \in [1, n] \end{array}}{U, I, \sigma \models \bot}$$

- Example: Suppose we have $S, \{x \mapsto a\} \models p(x)$ and $S, \{y \mapsto a\} \not\models p(y)$
- The proof rule for contradiction allows us to derive $\bot$

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x. p(x)) \rightarrow (\forall y. p(y))$$

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x.p(x)) \rightarrow (\forall y.p(y))$$

- We start by assuming it is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$.

1.	S, σ	$\not\models$	$(\forall x.p(x)) \rightarrow (\forall y.p(y))$	assumption

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x. p(x)) \rightarrow (\forall y. p(y))$$

- We start by assuming it is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$.

1.	S, σ	$\not\models$	$(\forall x. p(x)) \rightarrow (\forall y. p(y))$	assumption
2.	S, σ	$\models$	$\forall x. p(x)$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\forall y. p(y)$	1 and $\rightarrow$

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x.p(x)) \rightarrow (\forall y.p(y))$$

- We start by assuming it is not valid, i.e., there exists some S, σ such that $S, \sigma \;\not\models\; F$.

$$\begin{array}{llcll}
1. & S, \sigma & \not\models & (\forall x.p(x)) \rightarrow (\forall y.p(y)) & \text{assumption} \\
2. & S, \sigma & \models & \forall x.p(x) & \text{1 and } \rightarrow \\
3. & S, \sigma & \not\models & \forall y.p(y) & \text{1 and } \rightarrow \\
4. & S, \sigma[y \mapsto o] & \not\models & p(y) & \text{3 and } \not\models \forall x
\end{array}$$

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x.p(x)) \rightarrow (\forall y.p(y))$$

- We start by assuming it is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$.

$$\begin{array}{lrcll}
1. & S, \sigma & \not\models & (\forall x.p(x)) \rightarrow (\forall y.p(y)) & \text{assumption} \\
2. & S, \sigma & \models & \forall x.p(x) & 1 \text{ and } \rightarrow \\
3. & S, \sigma & \not\models & \forall y.p(y) & 1 \text{ and } \rightarrow \\
4. & S, \sigma[y \mapsto o] & \not\models & p(y) & 3 \text{ and } \not\models \forall x \\
5. & S, \sigma[x \mapsto o] & \models & p(x) & 2 \text{ and } \models \forall x
\end{array}$$

Example 1: Proving Validity

- Prove the validity of formula:

$$F : (\forall x.p(x)) \rightarrow (\forall y.p(y))$$

- We start by assuming it is not valid, i.e., there exists some S, σ such that $S, \sigma \not\models F$.

$$\begin{array}{llcll}
1. & S, \sigma & \not\models & (\forall x.p(x)) \rightarrow (\forall y.p(y)) & \text{assumption} \\
2. & S, \sigma & \models & \forall x.p(x) & \text{1 and } \rightarrow \\
3. & S, \sigma & \not\models & \forall y.p(y) & \text{1 and } \rightarrow \\
4. & S, \sigma[y \mapsto o] & \not\models & p(y) & \text{3 and } \not\models \ \forall x \\
5. & S, \sigma[x \mapsto o] & \models & p(x) & \text{2 and } \models \ \forall x \\
6. & S, \sigma & \models & \perp & \text{4,5}
\end{array}$$

Example 2

- Is this formula valid?

$$F : (\forall x.\ (p(x) \lor q(x))) \rightarrow (\exists x. p(x) \lor \forall x. q(x))$$

Example 2

- Is this formula valid? Yes!

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x. p(x) \vee \forall x. q(x))$$

Example 2

- Is this formula valid? Yes!

$$F : (\forall x.\ (p(x) \lor q(x))) \rightarrow (\exists x. p(x) \lor \forall x. q(x))$$

- Informal argument: Suppose $\forall x.(p(x) \lor q(x))$ holds

Example 2

- Is this formula valid? Yes!

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x. p(x) \vee \forall x. q(x))$$

- Informal argument: Suppose $\forall x.(p(x) \vee q(x))$ holds
- This means either $q(x)$ for all objects (i.e., $\forall x.q(x)$)

Example 2

- Is this formula valid? Yes!

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Informal argument: Suppose $\forall x.(p(x) \vee q(x))$ holds
- This means either $q(x)$ for all objects (i.e., $\forall x.q(x)$)
- Or if $q(x)$ does not hold for some object o, then $p(x)$ must hold for that object o (i.e, $\exists x.p(x)$

Example 2

- Is this formula valid? Yes!

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Informal argument: Suppose $\forall x.(p(x) \vee q(x))$ holds
- This means either $q(x)$ for all objects (i.e., $\forall x.q(x)$)
- Or if $q(x)$ does not hold for some object o, then $p(x)$ must hold for that object o (i.e, $\exists x.p(x)$
- Thus, antecedent implies $\exists p(x) \vee \forall x.q(x)$

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \lor q(x))) \rightarrow (\exists x. p(x) \lor \forall x. q(x))$$

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x. p(x) \vee \forall x. q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x. p(x) \vee \forall x. q(x)$	1 and $\rightarrow$

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \lor q(x))) \rightarrow (\exists x.p(x) \lor \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \lor q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \lor \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\lor$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\lor$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o
7.	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4 and $\not\models$ $\exists x$, any o

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o
7.	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4 and $\not\models$ $\exists x$, any o
8.	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \vee q(x)$	2 and $\models$ $\forall x$, any o

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o
7.	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4 and $\not\models$ $\exists x$, any o
8.	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \vee q(x)$	2 and $\models$ $\forall x$, any o
$9a$.	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	8 and $\vee$
$9b$.	$S, \sigma[x \mapsto o]$	$\models$	$q(x)$	8 and $\vee$

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o
7.	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4 and $\not\models$ $\exists x$, any o
8.	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \vee q(x)$	2 and $\models$ $\forall x$, any o
$9a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	8 and $\vee$
$9b.$	$S, \sigma[x \mapsto o]$	$\models$	$q(x)$	8 and $\vee$
$10a.$	S, σ	$\models$	$\perp$	7, 9a

Example 2, cont

- Let's now prove validity using semantic argument method

$$F : (\forall x.\ (p(x) \vee q(x))) \rightarrow (\exists x.p(x) \vee \forall x.q(x))$$

- Let's assume there is some S, σ that does not entail ϕ, and derive contradiction on all branches

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \vee q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$\exists x.p(x) \vee \forall x.q(x)$	1 and $\rightarrow$
4.	S, σ	$\not\models$	$\exists x.p(x)$	3 and $\vee$
5.	S, σ	$\not\models$	$\forall x.q(x)$	3 and $\vee$
6.	$S, \sigma[x \mapsto o]$	$\not\models$	$q(x)$	5 and $\not\models$ $\forall x$, fresh o
7.	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4 and $\not\models$ $\exists x$, any o
8.	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \vee q(x)$	2 and $\models$ $\forall x$, any o
$9a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	8 and $\vee$
$9b.$	$S, \sigma[x \mapsto o]$	$\models$	$q(x)$	8 and $\vee$
$10a.$	S, σ	$\models$	$\perp$	7, 9a
$10b.$	S, σ	$\models$	$\perp$	6, 9b

Example 3

- Is this formula valid?

$$F : (\forall x.p(x, x)) \rightarrow (\exists x.\forall y.p(x, y))$$

Example 3

- Is this formula valid? No!

$$F : (\forall x.p(x, x)) \rightarrow (\exists x.\forall y.p(x, y))$$

Example 3

- Is this formula valid? No!

$$F : (\forall x. p(x, x)) \rightarrow (\exists x. \forall y. p(x, y))$$

- Intuitively, antecedent says $p(o, o)$ holds for every object o

Example 3

- Is this formula valid? No!

$$F : (\forall x.p(x, x)) \rightarrow (\exists x.\forall y.p(x, y))$$

- Intuitively, antecedent says $p(o, o)$ holds for every object o
- Consequent says there exists some object, say o_1, for which $p(o_1, \_)$ holds

Example 3

- Is this formula valid? No!

$$F : (\forall x . p(x, x)) \rightarrow (\exists x . \forall y . p(x, y))$$

- Intuitively, antecedent says $p(o, o)$ holds for every object o
- Consequent says there exists some object, say o_1, for which $p(o_1, \_)$ holds
- Clearly, these mean very different things

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x,x)) \rightarrow (\exists x.\forall y.p(x,y))$$

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x, x)) \to (\exists x.\forall y.p(x, y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x. p(x, x)) \rightarrow (\exists x. \forall y. p(x, y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$
- In this case, σ not necessary since no free variables

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x,x)) \rightarrow (\exists x.\forall y.p(x,y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$
- In this case, σ not necessary since no free variables
- Choose $U = \{\star, \bullet\}$, and $I(p) = \{\langle \star, \star \rangle, \langle \bullet, \bullet \rangle\}$

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x,x)) \rightarrow (\exists x.\forall y.p(x,y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$
- In this case, σ not necessary since no free variables
- Choose $U = \{\star, \bullet\}$, and $I(p) = \{\langle\star,\star\rangle, \langle\bullet,\bullet\rangle\}$
- Clearly, under I, $\forall x.p(x,x)$ evaluates to true.

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x, x)) \to (\exists x.\forall y.p(x, y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$
- In this case, σ not necessary since no free variables
- Choose $U = \{\star, \bullet\}$, and $I(p) = \{\langle\star, \star\rangle, \langle\bullet, \bullet\rangle\}$
- Clearly, under I, $\forall x.p(x, x)$ evaluates to true.
- Furthermore, under I, $(\exists x.\forall y.p(x, y))$ evaluates to false.

Example 3, cont

- Now, how do we formally prove this formula is not valid?

$$F : (\forall x.p(x,x)) \rightarrow (\exists x.\forall y.p(x,y))$$

- We have to come up with U, I, σ such that $U, I, \sigma \not\models F$
- In this case, σ not necessary since no free variables
- Choose $U = \{\star, \bullet\}$, and $I(p) = \{\langle\star,\star\rangle, \langle\bullet,\bullet\rangle\}$
- Clearly, under I, $\forall x.p(x,x)$ evaluates to true.
- Furthermore, under I, $(\exists x.\forall y.p(x,y))$ evaluates to false.
- Thus, I is a falsfiying interpretation of F.

Example 4

- Is the following formula valid?

$$(\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

Example 4

- Is the following formula valid? Yes

$$(\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

Example 4

- Is the following formula valid? Yes

$$(\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Suppose $(\forall x.p(x) \wedge q(x))$ holds, we know $p(x)$ and $q(x)$ hold for every object o

Example 4

- Is the following formula valid? Yes

$$(\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Suppose $(\forall x.p(x) \land q(x))$ holds, we know $p(x)$ and $q(x)$ hold for every object o

- Thus, $p(x)$ must hold for every object (i.e., $\forall x.p(x)$) and $q(x)$ must hold for every object (i.e., $\forall x.q(x)$)

Example 4

- Is the following formula valid? Yes

$$(\forall x.(p(x) \land q(x))) \to (\forall x.p(x)) \land (\forall x.q(x))$$

- Suppose $(\forall x.p(x) \land q(x))$ holds, we know $p(x)$ and $q(x)$ hold for every object o
- Thus, $p(x)$ must hold for every object (i.e., $\forall x.p(x)$) and $q(x)$ must hold for every object (i.e., $\forall x.q(x)$)
- Thus, we also have $\forall x.p(x) \land \forall x.q(x)$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

1.	$S, \sigma \not\models F$	assumption

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \; (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

1.	S, σ	$\not\models$	F	assumption
2.	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\rightarrow$
3.	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\rightarrow$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\land$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \wedge q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \wedge (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\wedge$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\wedge$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\land$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\land$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F : \ (\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \wedge q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \wedge (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\wedge$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\wedge$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\land$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\land$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \land q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\land$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F : \ (\forall x.(p(x) \land q(x))) \to (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\to$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\to$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\land$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\land$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \land q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\land$
$8a.$	$S, \sigma[x \mapsto o]$	$\models$	$\perp$	5a, 7a

Example 4, cont

- Let's prove validity using semantic argument method:

$$F:\ (\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma\ \not\models\ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \wedge q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \wedge (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\wedge$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\wedge$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\wedge$
$8a.$	$S, \sigma[x \mapsto o]$	$\models$	$\perp$	5a, 7a
$5b.$	$S, \sigma[x \mapsto o']$	$\not\models$	$q(x)$	4b and $\forall$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F:\ (\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \not\models F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \wedge q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \wedge (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\wedge$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\wedge$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\wedge$
$8a.$	$S, \sigma[x \mapsto o]$	$\models$	$\perp$	5a, 7a
$5b.$	$S, \sigma[x \mapsto o']$	$\not\models$	$q(x)$	4b and $\forall$
$6b.$	$S, \sigma[x \mapsto o']$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \wedge q(x))) \rightarrow (\forall x.p(x)) \wedge (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \wedge q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \wedge (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\wedge$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\wedge$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\wedge$
$8a.$	$S, \sigma[x \mapsto o]$	$\models$	$\perp$	5a, 7a
$5b.$	$S, \sigma[x \mapsto o']$	$\not\models$	$q(x)$	4b and $\forall$
$6b.$	$S, \sigma[x \mapsto o']$	$\models$	$p(x) \wedge q(x)$	2 and $\forall$
$7b.$	$S, \sigma[x \mapsto o']$	$\models$	$q(x)$	6b and $\wedge$

Example 4, cont

- Let's prove validity using semantic argument method:

$$F: \ (\forall x.(p(x) \land q(x))) \rightarrow (\forall x.p(x)) \land (\forall x.q(x))$$

- Assume there is a S, σ such that $S, \sigma \ \not\models \ F$

$1.$	S, σ	$\not\models$	F	assumption
$2.$	S, σ	$\models$	$\forall x.(p(x) \land q(x))$	1 and $\rightarrow$
$3.$	S, σ	$\not\models$	$(\forall x.p(x)) \land (\forall x.q(x))$	1 and $\rightarrow$
$4a.$	S, σ	$\not\models$	$(\forall x.p(x))$	3 and $\land$
$4b.$	S, σ	$\not\models$	$(\forall x.q(x))$	3 and $\land$
$5a.$	$S, \sigma[x \mapsto o]$	$\not\models$	$p(x)$	4a and $\forall$
$6a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x) \land q(x)$	2 and $\forall$
$7a.$	$S, \sigma[x \mapsto o]$	$\models$	$p(x)$	6a and $\land$
$8a.$	$S, \sigma[x \mapsto o]$	$\models$	$\perp$	5a, 7a
$5b.$	$S, \sigma[x \mapsto o']$	$\not\models$	$q(x)$	4b and $\forall$
$6b.$	$S, \sigma[x \mapsto o']$	$\models$	$p(x) \land q(x)$	2 and $\forall$
$7b.$	$S, \sigma[x \mapsto o']$	$\models$	$q(x)$	6b and $\land$
$8b.$	$S, \sigma[x \mapsto o']$	$\models$	$\perp$	5b, 7b

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.
- Soundness: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.
- Soundness: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.
- Translation: The proof system does not reach wrong conclusions

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.
- Soundness: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.
- Translation: The proof system does not reach wrong conclusions
- Completeness: If formula F is valid, then there exists a finite-length proof in which every branch derives $\perp$

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.
- Soundness: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.
- Translation: The proof system does not reach wrong conclusions
- Completeness: If formula F is valid, then there exists a finite-length proof in which every branch derives $\perp$
- Translation: There are no valid first-order formulas which we cannot prove to be valid using our proof rules.

Soundness and Completeness of Proof Rules

- The proof rules we used are sound and complete.
- Soundness: If every branch of semantic argument proof derives a contradiction, then F is indeed valid.
- Translation: The proof system does not reach wrong conclusions
- Completeness: If formula F is valid, then there exists a finite-length proof in which every branch derives $\perp$
- Translation: There are no valid first-order formulas which we cannot prove to be valid using our proof rules.
- Completeness in this context also called refutational completeness

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)
- Review: A problem is decidable iff there exists a procedure P such that, for any input:

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)
- Review: A problem is decidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)
- Review: A problem is decidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. halts and says "no" if the answer is negative

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)
- Review: A problem is decidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. halts and says "no" if the answer is negative
- But, what about the completeness result? Doesn't this contradict undecidability?

Important Properties of First Order Logic

- Really important result: It is undecidable whether a first-order formula is valid. (Church and Turing)
- Review: A problem is decidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. halts and says "no" if the answer is negative
- But, what about the completeness result? Doesn't this contradict undecidability?
- No, because completeness says we will find proof of validity if it exists, but if formula is invalid, we might search forever.

Semidecidability of First-Order Logic

- First-order logic is semidecidable

Semidecidability of First-Order Logic

- First-order logic is semidecidable
- A decision problem is semidecidable iff there exists a procedure P such that, for any input:

Semidecidability of First-Order Logic

- First-order logic is semidecidable
- A decision problem is semidecidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive

Semidecidability of First-Order Logic

- First-order logic is semidecidable
- A decision problem is semidecidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. P may not terminate if the answer is negative

Semidecidability of First-Order Logic

- First-order logic is semidecidable
- A decision problem is semidecidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. P may not terminate if the answer is negative
- Thus, there exists an algorithm that always terminates and says if any arbitrary FOL formula is valid

Semidecidability of First-Order Logic

- First-order logic is semidecidable
- A decision problem is semidecidable iff there exists a procedure P such that, for any input:
  1. P halts and says "yes" if the answer is positive
  2. P may not terminate if the answer is negative
- Thus, there exists an algorithm that always terminates and says if any arbitrary FOL formula is valid
- But no algorithm is guaranteed to terminate if the FOL formula is not valid

Decidable Fragments of First-Order Logic

- Although full-first order logic is not decidable, there are fragments of FOL that are decidable.

Decidable Fragments of First-Order Logic

- Although full-first order logic is not decidable, there are fragments of FOL that are decidable.
- A fragment of FOL is a syntactially restricted subset of full FOL: e.g., no functions, or only universal quantifiers, etc.

Decidable Fragments of First-Order Logic

- Although full-first order logic is not decidable, there are fragments of FOL that are decidable.
- A fragment of FOL is a syntactially restricted subset of full FOL: e.g., no functions, or only universal quantifiers, etc.
- Some decidable fragments:
  - Quantifier-free first order logic
  - Monadic first-order logic
  - Bernays-Schönfinkel class

Quantifier-Free Fragment of FOL

- The quantifier-free fragment of FOL is the syntactically restricted subset of FOL where formulas do not contain universal or existential quantifiers.

Quantifier-Free Fragment of FOL

- The quantifier-free fragment of FOL is the syntactically restricted subset of FOL where formulas do not contain universal or existential quantifiers.
- Determining validity and satisfiability in quantifier-free FOL is decidable (NP-complete).

Quantifier-Free Fragment of FOL

- The quantifier-free fragment of FOL is the syntactically restricted subset of FOL where formulas do not contain universal or existential quantifiers.
- Determining validity and satisfiability in quantifier-free FOL is decidable (NP-complete).
- This fragment can be reduced to a theory we will explore later, theory of equality with uninterpreted functions

Monadic First-Order Logic

- Pure monadic FOL: all predicates are monadic (i.e., arity 1) and no function constants.

Monadic First-Order Logic

- Pure monadic FOL: all predicates are monadic (i.e., arity 1) and no function constants.
- Impure monadic FOL: both monadic predicates and monadic function constants allowed

Monadic First-Order Logic

- Pure monadic FOL: all predicates are monadic (i.e., arity 1) and no function constants.
- Impure monadic FOL: both monadic predicates and monadic function constants allowed
- Result: Monadic first-order logic is decidable (both versions)

Monadic First-Order Logic

- Pure monadic FOL: all predicates are monadic (i.e., arity 1) and no function constants.
- Impure monadic FOL: both monadic predicates and monadic function constants allowed
- Result: Monadic first-order logic is decidable (both versions)
- However, if we add even a single binary predicate, the logic becomes undecidable.

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:
  1. there are no function constants,

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:
  1. there are no function constants,
  2. only formulas of the form:

$$\exists x_1, \ldots, \exists x_n . \forall y_1, \ldots, \forall y_m . F(x_1, \ldots, x_n, y_1, \ldots y_m)$$

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:
  1. there are no function constants,
  2. only formulas of the form:

$$\exists x_1, \ldots, \exists x_n . \forall y_1, \ldots, \forall y_m . F(x_1, \ldots, x_n, y_1, \ldots y_m)$$

- Result: The Bernays-Schönfinkel fragment of FOL is decidable

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:
  1. there are no function constants,
  2. only formulas of the form:

$$\exists x_1, \ldots, \exists x_n . \forall y_1, \ldots, \forall y_m . F(x_1, \ldots, x_n, y_1, \ldots y_m)$$

- Result: The Bernays-Schönfinkel fragment of FOL is decidable
- Database query language Datalog is based on Bernays-Schönfinkel class of FOL

Bernays-Schönfinkel Class

- The Bernays-Schönfinkel class is a fragment of FOL where:
  1. there are no function constants,
  2. only formulas of the form:

$$\exists x_1, \ldots, \exists x_n. \forall y_1, \ldots, \forall y_m. F(x_1, \ldots, x_n, y_1, \ldots y_m)$$

- Result: The Bernays-Schönfinkel fragment of FOL is decidable
- Database query language Datalog is based on Bernays-Schönfinkel class of FOL
- However, it has additional restriction that all clauses are Horn clauses (i.e., at most one positive literal in each clause)

Datalog

- Datalog is a programming language that allows adding/querying facts in a deductive databases

Datalog

- Datalog is a programming language that allows adding/querying facts in a deductive databases
- An example Datalog program:

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

Datalog

- Datalog is a programming language that allows adding/querying facts in a deductive databases
- An example Datalog program:

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- Last statement is a query: Is there anyone in the database who is John's ancestor (and if so, who?)

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

$$parent(bill, mary) \land parent(mary, john) \land$$

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

$$\begin{array}{c} parent(bill, mary) \land parent(mary, john) \land \\ (\forall x, y.\ parent(x, y) \rightarrow ancestor(x, y)) \land \end{array}$$

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

$$\begin{gathered} parent(bill, mary) \wedge parent(mary, john) \wedge \\ (\forall x, y.\ parent(x, y) \rightarrow ancestor(x, y)) \wedge \\ (\forall x, y, z. parent(x, y) \wedge parent(y, z) \rightarrow ancestor(x, z)) \wedge \end{gathered}$$

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

$$\begin{gathered} parent(bill, mary) \land parent(mary, john) \land \\ (\forall x, y.\ parent(x, y) \rightarrow ancestor(x, y)) \land \\ (\forall x, y, z. parent(x, y) \land parent(y, z) \rightarrow ancestor(x, z)) \land \\ (\exists x. ancestor(x, john)) \end{gathered}$$

Datalog, cont.

```
parent(bill, mary).  % Bill is Mary's parent
parent(mary, john).  % Mary is John's parent

ancestor(X,Y) :- parent(X,Y).
ancestor(X,Z) :- parent(X,Y), ancestor(Y,Z).

?-ancestor(X, john).
```

- This program is just syntactic sugar for FOL:

$$\begin{gathered} parent(bill, mary) \land parent(mary, john) \land \\ (\forall x, y.\ parent(x, y) \to ancestor(x, y)) \land \\ (\forall x, y, z. parent(x, y) \land parent(y, z) \to ancestor(x, z)) \land \\ (\exists x. ancestor(x, john)) \end{gathered}$$

- Thus, if this formula is satisfiable, there is someone in our database who is John's ancestor

Datalog and Logic Programming Languages

- A Datalog interpreter is nothing more than a solver for Bernays-Schönfinkel fragment of FOL

Datalog and Logic Programming Languages

- A Datalog interpreter is nothing more than a solver for Bernays-Schönfinkel fragment of FOL
- Since this fragment is decidable, Datalog programs always terminate

Datalog and Logic Programming Languages

- A Datalog interpreter is nothing more than a solver for Bernays-Schönfinkel fragment of FOL
- Since this fragment is decidable, Datalog programs always terminate
- In general, interpreters for all logic programming languages decide satisfiability in FOL or a fragment

Datalog and Logic Programming Languages

- A Datalog interpreter is nothing more than a solver for Bernays-Schönfinkel fragment of FOL
- Since this fragment is decidable, Datalog programs always terminate
- In general, interpreters for all logic programming languages decide satisfiability in FOL or a fragment
- A popular logic programming language is Prolog

Datalog and Logic Programming Languages

- A Datalog interpreter is nothing more than a solver for Bernays-Schönfinkel fragment of FOL
- Since this fragment is decidable, Datalog programs always terminate
- In general, interpreters for all logic programming languages decide satisfiability in FOL or a fragment
- A popular logic programming language is Prolog
- Unlike Datalog, it is based on full FOL, so Prolog programs may not terminate

Compactness of First-Order Logic

- Another important property of FOL is compactness.

Compactness of First-Order Logic

- Another important property of FOL is compactness.
- A logic is called compact if an infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.

Compactness of First-Order Logic

- Another important property of FOL is compactness.
- A logic is called compact if an infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.
- Theorem (due to Gödel): First-order logic is compact.

Compactness of First-Order Logic

- Another important property of FOL is compactness.
- A logic is called compact if an infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.
- Theorem (due to Gödel): First-order logic is compact.
- Proof of compactness of FOL follows from the completeness of proof rules.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.
- Suppose FOL was not compact, i.e., there is an infinite set of sentences Γ that are unsat, but every finite subset Σ is sat.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.
- Suppose FOL was not compact, i.e., there is an infinite set of sentences Γ that are unsat, but every finite subset Σ is sat.
- By completeness of proof rules, if Γ is unsat, there exists a finite-length proof of unsatisfiability.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.
- Suppose FOL was not compact, i.e., there is an infinite set of sentences Γ that are unsat, but every finite subset Σ is sat.
- By completeness of proof rules, if Γ is unsat, there exists a finite-length proof of unsatisfiability.
- But this means the proof must use a finite subset of sentences Σ of Γ, otherwise proof could not be finite.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.
- Suppose FOL was not compact, i.e., there is an infinite set of sentences Γ that are unsat, but every finite subset Σ is sat.
- By completeness of proof rules, if Γ is unsat, there exists a finite-length proof of unsatisfiability.
- But this means the proof must use a finite subset of sentences Σ of Γ, otherwise proof could not be finite.
- But this implies there is also a proof of unsatisfiability of Σ.

Proof of Compactness

- Recall: Completeness means that if a formula is unsatisfiable, then there exists a finite-length proof of unsatisfiability.
- Suppose FOL was not compact, i.e., there is an infinite set of sentences Γ that are unsat, but every finite subset Σ is sat.
- By completeness of proof rules, if Γ is unsat, there exists a finite-length proof of unsatisfiability.
- But this means the proof must use a finite subset of sentences Σ of Γ, otherwise proof could not be finite.
- But this implies there is also a proof of unsatisfiability of Σ.
- Thus, by soundness of proof rules, Σ must be unsat. $\square$

Consequences of Compactness

- Proof of compactness might look like a useless property, but it has very interesting consequences!

Consequences of Compactness

- Proof of compactness might look like a useless property, but it has very interesting consequences!
- Compactness can be used to show that a variety of interesting properties are not expressible in first-order logic.

Consequences of Compactness

- Proof of compactness might look like a useless property, but it has very interesting consequences!
- Compactness can be used to show that a variety of interesting properties are not expressible in first-order logic.
- For instance, we can use compactness theorem to show that transitive closure is not expressible in first order logic.

Transitive Closure

- Given a directed graph $G = (V, E)$, the transitive closure of G is defined as the graph $G^* = (V, E^*)$ where:

$$E^* = \{(n, n') \mid \text{ if there is a path from vertex n to n}'\}$$

Transitive Closure

- Given a directed graph $G = (V, E)$, the transitive closure of G is defined as the graph $G^* = (V, E^*)$ where:

$$E^* = \{(n, n') \mid \text{ if there is a path from vertex n to n}'\}$$

- Observe: A binary predicate $p(t, t')$ be viewed as a graph containing an edge from node t to t'

Transitive Closure

- Given a directed graph $G = (V, E)$, the transitive closure of G is defined as the graph $G^* = (V, E^*)$ where:

$$E^* = \{(n, n') \mid \text{ if there is a path from vertex n to n}'\}$$

- Observe: A binary predicate $p(t, t')$ be viewed as a graph containing an edge from node t to t'

- Thus, the concept of transitive closure applies to binary predicates as well

Transitive Closure

- Given a directed graph $G = (V, E)$, the transitive closure of G is defined as the graph $G^* = (V, E^*)$ where:

$$E^* = \{(n, n') \mid \text{ if there is a path from vertex n to n}'\}$$

- Observe: A binary predicate $p(t, t')$ be viewed as a graph containing an edge from node t to t'
- Thus, the concept of transitive closure applies to binary predicates as well
- A binary predicate T is the transitive closure of predicate p if $\langle t_0, t_n \rangle \in T$ iff there exists some sequence $t_0, t_1 \ldots, t_n$ such that $\langle t_i, t_{i+1} \rangle \in p$

"Expressing" Transitive Closure in FOL

- At first glance, it looks like transitive closure T of binary relation p is expressible in FOL:

$$\forall x, \forall z.(T(x,z) \ \leftrightarrow \ (p(x,z) \vee \exists y.p(x,y) \wedge T(y,z)))$$

“Expressing” Transitive Closure in FOL

- At first glance, it looks like transitive closure T of binary relation p is expressible in FOL:

$$\forall x, \forall z.(T(x,z) \;\leftrightarrow\; (p(x,z) \vee \exists y. p(x,y) \wedge T(y,z)))$$

- But this formula does not describe transitive closure at all!

"Expressing" Transitive Closure in FOL

- At first glance, it looks like transitive closure T of binary relation p is expressible in FOL:

$$\forall x, \forall z.(T(x,z) \;\leftrightarrow\; (p(x,z) \lor \exists y. p(x,y) \land T(y,z)))$$

- But this formula does not describe transitive closure at all!

- To see why, consider $U = \mathbb{N}$, p is equality predicate, and T is relation that is true for *any* number x, y.

"Expressing" Transitive Closure in FOL

- At first glance, it looks like transitive closure T of binary relation p is expressible in FOL:

$$\forall x, \forall z.(T(x,z) \ \leftrightarrow \ (p(x,z) \vee \exists y.p(x,y) \wedge T(y,z)))$$

- But this formula does not describe transitive closure at all!
- To see why, consider $U = \mathbb{N}$, p is equality predicate, and T is relation that is true for *any* number x, y.
- Clearly, this T is not the transitive closure of equality, but this structure is actually a model of the formula.

"Expressing" Transitive Closure in FOL

- At first glance, it looks like transitive closure T of binary relation p is expressible in FOL:

$$\forall x, \forall z.(T(x,z) \;\leftrightarrow\; (p(x,z) \vee \exists y.p(x,y) \wedge T(y,z)))$$

- But this formula does not describe transitive closure at all!
- To see why, consider $U = \mathbb{N}$, p is equality predicate, and T is relation that is true for *any* number x, y.
- Clearly, this T is not the transitive closure of equality, but this structure is actually a model of the formula.
- Thus, the formula above is not a definition of transitive closure at all!

Transitive Closure and FOL

- In fact, no matter how hard we try to correct this definition, we cannot express transitive closure in FOL

Transitive Closure and FOL

- In fact, no matter how hard we try to correct this definition, we cannot express transitive closure in FOL
- Will use compactness theorem to show that transitive closure is not expressible in FOL

Transitive Closure and FOL

- In fact, no matter how hard we try to correct this definition, we cannot express transitive closure in FOL
- Will use compactness theorem to show that transitive closure is not expressible in FOL
- Compactness: An infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.

Transitive Closure and FOL

- In fact, no matter how hard we try to correct this definition, we cannot express transitive closure in FOL
- Will use compactness theorem to show that transitive closure is not expressible in FOL
- Compactness: An infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.
- For contradiction, suppose transitive closure is expressible in first order logic

Transitive Closure and FOL

- In fact, no matter how hard we try to correct this definition, we cannot express transitive closure in FOL
- Will use compactness theorem to show that transitive closure is not expressible in FOL
- Compactness: An infinite set of sentences Γ is satisfiable iff every finite subset of Γ is satisfiable.
- For contradiction, suppose transitive closure is expressible in first order logic
- Let Γ be a (possibly infinite) set of sentences expressing that T is the transitive closure of p.

Proof I

- $\Psi^n(a, b)$ encode the proposition: there is no path of length n from a to b.

Proof I

- $\Psi^n(a, b)$ encode the proposition: there is no path of length n from a to b.
- In particular, $\Psi^1 = \neg p(a, b)$

Proof I

- $\Psi^n(a, b)$ encode the proposition: there is no path of length n from a to b.
- In particular, $\Psi^1 = \neg p(a, b)$
- Similarly,

$$\Psi^n = \neg \exists x_1, \ldots . x_{n-1}.(p(a, x_1) \wedge p(x_1, x_2) \wedge \ldots \wedge p(x_{n-1}, b))$$

Proof II

- Recall: Γ is a set of propositions encoding T is transitive closure of p.

Proof II

- Recall: Γ is a set of propositions encoding T is transitive closure of p.
- Now, construct Γ' as follows:

$$\Gamma' = \Gamma \cup \{ T(a, b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

Proof II

- Recall: Γ is a set of propositions encoding T is transitive closure of p.
- Now, construct Γ' as follows:

$$\Gamma' = \Gamma \cup \{ T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Observe: Γ' is unsatisfiable because:

Proof II

- Recall: Γ is a set of propositions encoding T is transitive closure of p.
- Now, construct Γ' as follows:

$$\Gamma' = \Gamma \cup \{\, T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Observe: Γ' is unsatisfiable because:

  1. Since Γ encodes that T is transitive closure of p, $T(a,b)$ says there is some path from a to b

Proof II

- Recall: Γ is a set of propositions encoding T is transitive closure of p.
- Now, construct Γ' as follows:

$$\Gamma' = \Gamma \cup \{ T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Observe: Γ' is unsatisfiable because:
  1. Since Γ encodes that T is transitive closure of p, $T(a,b)$ says there is some path from a to b
  2. The infinite set of propositions $\Psi^1, \Psi^2, \ldots$ say that there is no path of any length from a to b

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{\, T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a, b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.

- Observe: This finite subset is satisfied by a model where there is a path of length i from a to b

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a, b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.
- Observe: This finite subset is satisfied by a model where there is a path of length i from a to b
- Thus, every finite subset of Γ' is satisfiable.

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a, b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.
- Observe: This finite subset is satisfied by a model where there is a path of length i from a to b
- Thus, every finite subset of Γ' is satisfiable.
- By the compactness theorem, this would imply Γ' is also satisfiable

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.
- Observe: This finite subset is satisfied by a model where there is a path of length i from a to b
- Thus, every finite subset of Γ' is satisfiable.
- By the compactness theorem, this would imply Γ' is also satisfiable
- But we just showed that Γ' is unsatisfiable!

Proof III

- Now, consider any finite subset of Γ':

$$\Gamma' = \Gamma \cup \{ T(a,b), \Psi^1, \Psi^2, \Psi^3, \ldots, \}$$

- Clearly, any finite subset does not contain Ψ_i for some i.
- Observe: This finite subset is satisfied by a model where there is a path of length i from a to b
- Thus, every finite subset of Γ' is satisfiable.
- By the compactness theorem, this would imply Γ' is also satisfiable
- But we just showed that Γ' is unsatisfiable!
- Thus, transitive closure cannot be expressed in FOL!

Summary

- Semantic argument method for proving validity in FOL

Summary

- Semantic argument method for proving validity in FOL
- Soundness and completeness of semantic argument method

Summary

- Semantic argument method for proving validity in FOL
- Soundness and completeness of semantic argument method
- Important properties of FOL: undecidability, semidecidability, compactness

Summary

- Semantic argument method for proving validity in FOL
- Soundness and completeness of semantic argument method
- Important properties of FOL: undecidability, semidecidability, compactness
- Compactness: useful to show what is not expressible in FOL

Summary

- Semantic argument method for proving validity in FOL
- Soundness and completeness of semantic argument method
- Important properties of FOL: undecidability, semidecidability, compactness
- Compactness: useful to show what is not expressible in FOL
- Next lecture: Basics of automated first-order theorem provers (much less theoretical)