TF: First Order Logic (AID-M)

0 - An introduction for First Order Logic

Markus Mayer, Cezar Ionescu
markus.mayer@th-deg.de

References

- Barwise and Etchemendy, *Language, Proof and Logic*, 2nd Ed 2011
- Older editions under a different name (*The Language of First-Order Logic*, e.g. 3rd Ed 1992) are just as useful
- Hint: https://www.edx.org/course/language-proof-and-logic is free and you get a copy of the latest edition!
- The slides were originally created by Prof. Ionescu and adapted by Prof. Mayer.

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\wedge$, $\vee$, $\neg$, $\to$, $\leftrightarrow$

Quantifiers

Logic

First-order logic (also: *first-order predicate logic*, FOL):

- FOL is a **family** of languages, having the same "grammar" and sharing important vocabulary elements (connectives and quantifiers).
- The languages in this family differ in the specific vocabulary used to form their most basic sentences, the **atomic sentences**.
- Atomic sentences correspond to the simplest English sentences: some names connected by a predicate ($2 < 3$, $\mathit{Tall}(\mathit{Sandy})$, $\mathit{LeftOf}(\mathit{Sarah}, \mathit{Chris})$).
- Different versions of FOL have different **names** and **predicates** available.

Individual constants

- The sentences of a first-order language are about objects in a **domain of discourse** or **of quantification** (also **universe** of discourse), which is **not empty**
- **Individual constants** (also *names*) are symbols used to refer to some **fixed** individual object.
- Thus, individual constants are similar to names in ordinary language, but they refer to **exactly one** object.
- An object may have more than one name, or no name at all.
- Examples: *(what are the appropriate domains of discourse?)*
  - $2, 10, \pi$
  - `Anne`, `Bob`
  - `Prolog`, `C`, ...

Predicate symbols

- **Predicate symbols** are used to express some property of objects or some relation between objects.
- Each predicate symbol comes with a single, fixed **arity**: a number that tells us how many names it needs in order to form a sentence.
- Examples:
  - $\mathtt{even}$ has arity 1
  - $=$ has arity 2
  - $\mathtt{between}$ ("B is between A and C") has arity 3
- *We assume* that the properties or relations expressed by the predicate symbols are **determinate**, i.e., that it is a definite fact of the matter whether or not the objects have the property. *(What could be a counter-example?)*

Function symbols and terms

- Function symbols allow us to form name-like terms from names and other name-like terms.
- Each function symbol has an *arity*.
- Simple terms: individual constants
- Complex terms: a function symbol of arity n followed by n terms
- A term refers to *one and only one* object.
- Examples:

Function symbol	Arity	Term
`mother`	1	`mother(Anne)`
$+$	2	$10 + \pi$
`weight`	1	`weight(Bob)`

Problem: what does $weight(weight(Bob))$ refer to?

Atomic sentences

- An **atomic sentence** is formed by a predicate symbol followed by the respective number of terms.
- Sentences make **claims** (*express propositions*).
- Since predicates express determinate properties and terms denote definite individuals, it follows that sentences express claims that are either **true** or **false**.
- Examples:
  - $2 < 3, 10 = \pi$
  - `atHome(Anne)`
  - `olderThan(Bob, Clara)`
  - `fatherOf(Dennis, Elektra)`

Convention

- `olderThan(Bob, Clara)`, `fatherOf(Dennis, Elektra)`

In the previous examples, we have used the ordinary convention: predicates start with lowercase, names with uppercase.
In our (!) FOL system, the notation is exactly the other way around:

- `OlderThan(bob, clara)`, `FatherOf(dennis, elektra)`

Other systems use different conventions (e.g., in Prolog both predicate symbols and names are written in lowercase).

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\land$, $\lor$, $\neg$, $\rightarrow$, $\leftrightarrow$

Quantifiers

Proofs

- A **proof** is a step-by-step demonstration that a conclusion S follows from some premises $P_1, \ldots, P_n$, i.e., that S **must** be true whenever the premises are all true.
- A **formal system** uses a **fixed set of rules** specifying what counts as an **acceptable step** in a proof.
- These rules **formalize** the methods of proof that we use informally in mathematics, science, and everyday life, in which we make essential use of the **meaning** of the statements.
- We'll use the **Fitch notation** for our proofs: Fitch notation on Wikipedia

Formal proofs

A formal proof will look something like this:

$$
\begin{array}{r|ll}
1 & P_1 & \\
2 & \vdots & \\
3 & P_n & \\
\cline{2-2}
4 & S_1 & \textit{justification} \\
5 & \vdots & \\
6 & S_m & \textit{justification} \\
7 & S & \textit{justification}
\end{array}
$$

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$

Quantifiers

Reiteration

$$\begin{array}{r|ll} i & P & \\ \vdots & \vdots & \\ \triangleright\ j & P & \textit{R: i} \end{array}$$

- R stands for **reiteration**.
- Justification: P must be true whenever P is true.
- This rule is useful in longer proofs, for example in order to recall a premise.

Identity introduction

$$\begin{array}{llll} & \vdots & \vdots & \\ \triangleright & \text{i.} & n = n & =I \\ & \vdots & \vdots & \end{array}$$

- Justification: the statement $n = n$ is always true.

Identity elimination

$$
\begin{array}{llll}
 & . & . & \\
 & i. & P(a) & \\
 & . & . & \\
 & . & . & \\
 & j. & a = b & \\
 & . & . & \\
 & . & . & \\
\triangleright & k. & P(b) & \mathit{=E: i, j}
\end{array}
$$

- If $a = b$, then a and b denote **the same object**, thus everything that applies to a will also apply to b.
- Compare this interpretation of = with the one found in various programming languages (C, Python, Java, ...): `x = x + 1`

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$

Quantifiers

Symmetry of identity

Give a proof of the following argument:

1	$a = b$	
2	???	*justification*
3	$b = a$	*justification*

Symmetry of identity (solution)

1	$a = b$	
2	$a = a$	=I
3	$b = a$	=E: 2, 1

What is P in the justification in line three?

Transitivity of identity

Give a proof of the following argument:

$$
\begin{array}{r|l}
1 & a = b \\
2 & b = c \\
\hline
3 & a = c
\end{array}
$$

Transitivity of identity

1	$a = b$	
2	$b = c$	
3	$a = c$	$=$E: 1, 2

What is P in the justification in line three?

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$

Quantifiers

Notation

- We will use P, Q, etc. as symbols for *sentences*.
- Please do not confuse the sentence P with the predicate P!
  - E.g., P(a, b) is a sentence, if P is a predicate with arity 2.
- It should always be clear from the context, which kind of logical element a symbol denotes.

Conjunction: $\wedge$

- If P and Q are sentences of FOL, then so is $P \wedge Q$
- The sentence $P \wedge Q$ is true **if and only if** both P and Q are true.

Truth table for $\wedge$

P	Q	$P \wedge Q$
F	F	F
F	T	F
T	F	F
T	T	T

Proof rules for $\wedge$

Conjunction introduction

$$
\begin{array}{llll}
 & . & . & \\
 & i. & P & \\
 & . & . & \\
 & . & . & \\
 & j. & Q & \\
 & . & . & \\
 & . & . & \\
\triangleright & k. & P \wedge Q & \wedge I: i, j
\end{array}
$$

Why is this a valid rule?

Proof rules for $\wedge$

Conjunction elimination

	.	.	
	i.	$P \wedge Q$	
	.	.	
	.	.	
$\triangleright$	j.	P	$\wedge E_l$: i

	.	.	
	i.	$P \wedge Q$	
	.	.	
	.	.	
$\triangleright$	j.	Q	$\wedge E_r$: i

Why is this a valid rule?

Example: Proof rules for $\wedge$

Prove that $B \wedge A$ follows from $A \wedge B$.

Example: Proof rules for $\wedge$ (solution)

Prove that B $\wedge$ A follows from A $\wedge$ B.

1	$A \wedge B$	
2	B	$\wedge E_r$: 1
3	A	$\wedge E_l$: 1
4	$B \wedge A$	$\wedge$I: 2, 3

Example: Proof rules for $\wedge$ (solution 2)

Prove that $B \wedge A$ follows from $A \wedge B$.

Semantic proof using truth tables:

A	B	$A \wedge B$	$B \quad A$
F	F	F	F
F	T	F	F
T	F	F	F
T	T	T	T

Example: Proof rules for $\wedge$

Show that $C \wedge B$ follows from $(A \wedge B) \wedge C$.

Example: Proof rules for ∧ (solution)

Show that C ∧ B follows from (A ∧ B) ∧ C.

1	$(A \wedge B) \wedge C$	
2	C	$\wedge E_r$: 1
3	$A \wedge B$	$\wedge E_l$: 1
4	B	$\wedge$E: 3
5	$C \wedge B$	$\wedge$I: 2, 4

Example: Proof rules for $\wedge$ (solution 2)

Show that $C \wedge B$ follows from $(A \wedge B) \wedge C$.
Using a truth table:

A	B	C	$A \wedge B$	$(A \wedge B) \wedge C$	$C \wedge B$
F	F	F	F	F	F
F	F	T	F	F	F
F	T	F	F	F	F
F	T	T	F	F	T
T	F	F	F	F	F
T	F	T	F	F	F
T	T	F	T	F	F
T	T	T	T	T	T

Remarks

- The truth table method of proof is *semantic*: It uses the *meaning* of “logical consequence”.
- Formal proofs are:
  - more flexible.
  - can be used in more complex cases.
  - can often impart an understanding of the situation.
- However:
  - The truth table method is simpler to implement.
  - Finding formal proofs requires an element of creativity.

Example: Proof rules for $\wedge$

Example: From $a = b \wedge b = c$ prove $a = c$.

1	$a = b \wedge b = c$	
2	$a = c$	*justification*

Example: Proof rules for $\land$ (solution)

Example: From $\mathtt{a = b \land b = c}$ prove $\mathtt{a = c}$.

1	$a = b \land b = c$	
2	$a = b$	$\land E_l$: 1
3	$b = c$	$\land E_r$: 1
4	$a = c$	$=$E: 2, 3

What property $P(x)$ was used in step 4?

Disjunction: $\vee$

- If P and Q are sentences of FOL, then so is $P \vee Q$.
- The sentence $P \vee Q$ is true exactly when **at least** one of the sentences P and Q is true.

Truth table for $\vee$

P	Q	$P \vee Q$
F	F	F
F	T	T
T	F	T
T	T	T

Proof rules for $\vee$

Introduction rules for $\vee$:

	$\vdots$	$\vdots$	
	i.	P	
	$\vdots$	$\vdots$	
$\triangleright$	j.	$P \vee Q$	$\vee I_l$: i

	$\vdots$	$\vdots$	
	i.	Q	
	$\vdots$	$\vdots$	
$\triangleright$	j.	$P \vee Q$	$\vee I_r$: i

Why are these rules valid?

Proof rules for $\vee$

Disjunction elimination (**proof by cases**):

i	$P \vee Q$	
$\dots$	$\dots$	
i_1	P	Assumption
$\dots$	$\dots$	
i_k	R	
$\dots$	$\dots$	
j_1	Q	Assumption
$\dots$	$\dots$	
j_l	R	
$\dots$	$\dots$	
k	R	$\vee$E: i, i_1–i_k, j_1–j_l

Important: an assumption can only be used inside its "box"!

Example: Proof rules for $\vee$

From $(B \wedge A) \vee (A \wedge C)$ deduce A.

Example: Proof rules for $\vee$ (solution)

From $(B \wedge A) \vee (A \wedge C)$ deduce A.

1	$(B \wedge A) \vee (A \wedge C)$	
2	$\quad B \wedge A$	
3	$\quad A$	$\wedge E_r$: 2
4	$\quad A \wedge C$	
5	$\quad A$	$\wedge E_l$: 4
6	A	$\vee$E: 1, 2–3, 4–5

Check this with a proof checker: https://proofs.openlogicproject.org

Distributivity

Show that

- $A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)$
- $A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C)$

(Without solution on the slides)

Negation: $\neg$

- If P is an FOL sentence, then so is $\neg P$
- $\neg P$ is true exactly when P is **not** true.
- A **literal** is either an atomic sentence, or the negation of an atomic sentence.

Truth table for $\neg$

P	$\neg P$
F	T
T	F

Contradiction: $\perp$

Before introducing the proof rules for negation, it is useful to introduce $\perp$.

- $\perp$ is a sentence of FOL
- $\perp$ cannot be true **under any circumstances**

Truth table for $\perp$

$\perp$
F

Proof rules for $\perp$

$\perp$ Introduction:

	.	.	
	i.	P	
	.	.	
	j.	$\neg P$	
	.	.	
$\triangleright$	k.	$\perp$	$\perp I: \ i, j$

Why is this rule valid?

Proof rules for $\perp$

	.	.	
	i.	$\perp$	
	.	.	
$\triangleright$	j.	P	$\perp E:\ i$

"Justification": $\perp$ cannot be true under any circumstances, so if you believe that $\perp$ is true, then you'll believe anything!
Why is this a valid principle?

Proof rules for ¬

¬ introduction:

$\ldots$	$\ldots$	
i	$\quad P$ (assumption)	
$\ldots$	$\quad \ldots$	
j	$\quad \bot$	
k	$\neg P$	$\neg$I: i–j

Justification: the assumption P has led to a contradiction, therefore it must be false.

Example

$\neg P \wedge \neg Q$ follows from $\neg(P \vee Q)$

$\neg(P \vee Q) \vdash \neg P \wedge \neg Q$

Example (Solution)

1	$\neg(P \vee Q)$	
2	$\quad P$	
3	$\quad P \vee Q$	$\vee I_l$: 2
4	$\quad \bot$	$\bot$I: 3, 1
5	$\neg P$	$\neg$I: 2–4
6	$\dots$	
9	$\neg Q$	*Exercise!*
10	$\neg P \wedge \neg Q$	$\wedge$I: 5, 9

Proof rules for $\neg$

$\neg$ elimination:

$$
\begin{array}{l|ll}
i & \neg\neg P & \\
\dots & \dots & \\
j & P & \neg\text{E: } i
\end{array}
$$

Why is this a valid rule?

Example

Show that $P \vee \neg P$ is a **logical truth** (can be derived with no premises!).

Example (Solution)

1	$\neg(P \vee \neg P)$	
2	P	
3	$P \vee \neg P$	$\vee I_l$: 2
4	$\bot$	$\bot$I: 3, 1
5	$\neg P$	$\neg$I: 2–4
6	$P \vee \neg P$	$\vee I_r$: 5
7	$\bot$	$\bot$I: 6, 1
8	$\neg\neg(P \vee \neg P)$	$\neg$I: 1–7
9	$P \vee \neg P$	$\neg$E: 8

THE CONDITIONAL: $\rightarrow$

- If P and Q are sentences of FOL, then so is $P \rightarrow Q$
- $P \rightarrow Q$ is false if and only if P is true and Q is false, otherwise it is true
- Why bother? Because $P \vdash Q$ if and only if $P \rightarrow Q$ is a logical truth.

Truth table for $\rightarrow$

P	Q	$P \rightarrow Q$
F	F	T
F	T	T
T	F	F
T	T	T

PROOF RULES FOR $\rightarrow$

$\rightarrow$ Introduction

$\dots$	$\dots$		
i		P	
$\dots$		$\dots$	
j		Q	
k	$P \rightarrow Q$		$\rightarrow$I: i–j

Why is this a valid rule?

Proof rules for $\to$

$\to$ Elimination

$$
\begin{array}{llll}
 & \vdots & \vdots & \\
 & i. & P \to Q & \\
 & \vdots & \vdots & \\
 & j. & P & \\
 & \vdots & \vdots & \\
\triangleright & k. & Q & \to E\text{: i, j}
\end{array}
$$

Why is this a valid rule?

Example

$\vdash A \to \neg\neg A$

Example (Solution)

$\vdash A \to \neg\neg A$

1	A	
2	$\neg A$	
3	A	R: 1
4	$\bot$	$\bot$I: 3, 2
5	$\neg\neg A$	$\neg$I: 2–4
6	$A \to \neg\neg A$	$\to$I: 1–5

Biconditional: $\leftrightarrow$

- If P and Q are sentences of FOL, then so is $P \leftrightarrow Q$.
- $P \leftrightarrow Q$ is true if and only if P and Q have the same truth value (either both are true, or both are false).

Truth table for the biconditional

P	Q	$P \leftrightarrow Q$
F	F	T
F	T	F
T	F	F
T	T	T

Remark: $P \leftrightarrow Q$ has the same truth table as $P \rightarrow Q \wedge Q \rightarrow P$.

Proof rules for ↔

↔ Introduction

...	...	
i	$\quad$ P (assumption)	
...	$\quad$ ...	
j	$\quad$ Q	
k	$\quad$ Q (assumption)	
...	$\quad$ ...	
l	$\quad$ P	
...	...	
m	$P \leftrightarrow Q$	$\leftrightarrow$I: i–j, m–l

Why is this a valid rule?

Proof rules for $\leftrightarrow$

Elimination rules for $\leftrightarrow$:

	.	.	
	i.	$P \leftrightarrow Q$	
	.	.	
	.	.	
	j.	P	
	.	.	
	.	.	
$\triangleright$	k.	Q	$\leftrightarrow E_l$: i, j

	.	.	
	i.	$P \leftrightarrow Q$	
	.	.	
	.	.	
	j.	Q	
	.	.	
	.	.	
$\triangleright$	k.	P	$\leftrightarrow E_r$: i, j

Remark: Most systems do not differentiate between $\leftrightarrow E_l$ and $\leftrightarrow E_r$.

More Exercises

We finished propositional (truth-functional) logic, and now we want to extend it to full first order logic.

Before that, we do some of the propositional logic exercises from our exercise sheet.

Contents

Introduction to Formal Logic

Formal Proofs

First Rules

First Formal Proofs

The Boolean Connectives: $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$

Quantifiers

Quantifiers

- Quantifiers allow a finer-grained analysis of statements
- In English: *determiners*
  - *every*, *some*, *most*, *the*, ...

Variables

- A new type of term: **variable**
- Variables are usually denoted with lowercase letters from the end of the English alphabet (u, v, w, x, y, z), with or without a numerical subscript
- A variable may be used where a term (e.g., a name) may be used.
    - E.g., $Likes(abby, x)$, $mother(x)$

Example: The FOL of arithmetic

The first-order language of arithmetic is given by

- two names: 0 and 1
- two *infix* binary (i.e., arity 2) function symbols: $+$ and $\times$
- one binary (arity 2) predicate: $<$.

Like **every FOL**, the language of arithmetic contains

- binary identity predicate: $=$

Example: terms in the FOL of arithmetic

- $(0 + 0)$, $(1 + (1 \times (0 + 1)))$, $(1 + (1 + 1)) \times (1 + 1)$

How can we express the fact that 3 is less than 4?

The FOL of arithmetic

Having variables, we can now form new terms such as .B., $(x + (1 \times y))$.

But these new terms are **not** names.

We can also form new sentence-like *formulas*, such as $x < (1 + (1 + 1))$ or $((1 + 1) \times (1 + 1)) = (x \times y)$

But these new formulas are **not** sentences.

Rules for terms (without variables)

1. If c is a name, then c is a term.
2. If f is a function symbol with arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.
3. Nothing else is a term.

Rules for terms (including variables)

1. If c is a name, then c is a term.
2. If x is a variable, then x is a term.
3. If f is a function symbol with arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.
4. Nothing else is a term.

Well-formed formulas

`Likes(dana, cody)` is an atomic sentence.

`Likes(x, y)` ist **not a sentence** (atomic or otherwise). It is **neither true, nor false**.

Well-formed formulas

1. If P is a predicate symbol of arity n, and t_1, ..., t_n are terms, then $P(t_1, ..., t_n)$ is a **wff**.
2. If P is a wff, then so is $\neg P$.
3. If P and Q are wffs, then so are $P \wedge Q$, $P \vee Q$, $P \rightarrow Q$, $P \leftrightarrow Q$.
4. If P is a wff and x is a variable, then $\forall x\ P$ is a wff and every occurence of x in $\forall x\ P$ is said to be **bound**.
5. If P is a wff and x is a variable, then $\exists x\ P$ is a wff and every occurence of x in $\exists x\ P$ is said to be **bound**.
6. Nothing else is a wff.

Variables that are not bound are said to be **free**.

Examples

- `Square(a) ∧ Smaller(a, b)`
- `Square(a) ∧ Smaller(a, x)`
  - x is
- `∃x (Square(a) ∧ Smaller(a, x))`
  - x is

Examples

- `Square(a) ∧ Smaller(a, b)`
- `Square(a) ∧ Smaller(a, x)`
  - x is free
- `∃x (Square(a) ∧ Smaller(a, x))`
  - x is

Examples

- `Square(a) ∧ Smaller(a, b)`
- `Square(a) ∧ Smaller(a, x)`
  - x is free
- `∃x (Square(a) ∧ Smaller(a, x))`
  - x is bound

Sentences

A **sentence** is a wff in which no variables occur free.

- `Square(a) ∧ Smaller(a, b)`
- `Square(a) ∧ Smaller(a, x)`
- `∃x (Square(a) ∧ Smaller(a, x))`

Which one of these are sentences?

Sentences

A **sentence** is a wff in which no variables occur free.

- `Square(a) ∧ Smaller(a, b)`
- `Square(a) ∧ Smaller(a, x)`
- `∃x (Square(a) ∧ Smaller(a, x))`

Which one of these are sentences?

The first and third examples are sentences.

The universal quantifier

- Quantified sentences make claims about some non-empty intended domain of discourse.
- `∀x P(x)` means that **every** object in the domain of discourse has the property P.
- Example:
  `∀x ((Prime(x) ∧ Even(x)) -> x = 2)`
- If the domain of discourse is finite, and if all the objects are named by $a_1, \dots, a_n$, then
  $\forall x\ P(x) \leftrightarrow P(a_1) \land \dots \land P(a_n)$

Proof rules for ∀

Elimination rule for ∀:

$$
\begin{array}{r|ll}
i & \forall x P(x) & \\
\vdots & \vdots & \\
j & P(c) & \qquad\qquad \forall\text{E: } i
\end{array}
$$

where c is a name (i.e., a term without variables).

Why is this a valid rule?

Example

Proof:

$$
\begin{array}{r|ll}
1 & \forall x \neg Likes(x, x) & \\
\hline
2 & \neg Likes(dana, dana) & \quad \textit{justification}
\end{array}
$$

Example (Solution)

1	$\forall x \neg Likes(x, x)$	
2	$\neg Likes(dana, dana)$	$\forall$E: 1

Proof rules for ∀

Introduction rule for ∀:

$$
\begin{array}{r|l|l}
i & & c\text{ “fresh”} \\
\vdots & & \vdots \\
j & & P(c) \\
\vdots & \vdots & \\
\color{blue}{k} & \forall x P(x) & \qquad\qquad \forall\text{I: } i\text{–}j
\end{array}
$$

Why is this a valid rule?

Proof rules for ∀

Introduction rule for ∀:

$$
\begin{array}{l|l|l l}
i & & c \text{ ``fresh''} & \\
\cline{3-3}
\vdots & & \vdots & \\
j & & P(c) & \\
\vdots & \vdots & & \\
k & \forall x P(x) & & \forall\mathrm{I}\text{: } i\text{–}j
\end{array}
$$

Why is this a valid rule?

If we can show that $P(c)$ holds for an *arbitrary* c, then it must hold for all objects. The introduction of a "fresh" name guarantees that no special features are assumed for this arbitrary object.

Example

Proof:

1	$\forall x(P(x) \wedge Q(x))$
2	$\forall x Q(x)$

Example (Solution)

1	$\forall x(P(x) \land Q(x))$	
2	$\quad c$	
3	$\quad P(c) \land Q(c)$	$\forall$E: 1
4	$\quad Q(c)$	$\land$E: 3
5	$\forall x Q(x)$	$\forall$I: 2–4

The existential quantifier

- $\exists x\ P(x)$ states that **at least** one object in the domain of discourse has the property P.
- Example:
  $\exists x\ (Prime(x) \land Even(x))$
- If the domain of discourse is finite, and if all the objects are named by $a_1, \ldots, a_n$, then
  $\exists x\ P(x)$ *leftrightarrow* $P(a_1) \lor \ldots \lor P(a_n)$

Proof rules for ∃

Introduction rule for ∃:

$$
\begin{array}{r|ll}
i & P(c) & \\
\vdots & \vdots & \\
j & \exists x P(x) & \qquad \exists\text{I}: i
\end{array}
$$

Why is this a valid rule?

Example

Example:

1	$Likes(dana, cody)$	
2	$\exists x Likes(x, cody)$	*justification*

Example (Solution)

Example:

1	$Likes(dana, cody)$	
2	$\exists x Likes(x, cody)$	$\exists$I: 1

Proof rules for ∃

Elimination rule for ∃:

i	$\exists x P(x)$		
$\vdots$	$\vdots$		
j		c, $P(c)$	c is “fresh”
$\vdots$		$\vdots$	
k		Q	c may not occur in Q!
$\vdots$	$\vdots$		
l	Q		∃I: i, j–k

Why is this a valid rule?

Example

Proof:

$$
\begin{array}{r|l}
1 & \exists x(P(x) \wedge Q(x)) \\
\hline
2 & \exists x Q(x)
\end{array}
$$

Example solution

1	$\exists x(P(x) \wedge Q(x))$	
2	$\quad c,\ P(c) \wedge Q(c)$	
3	$\quad Q(c)$	$\wedge$E: 2
4	$\quad \exists x Q(x)$	$\exists$I: 3
5	$\exists x Q(x)$	$\exists$E: 1, 2–4

Example: DeMorgan's laws 1

$\neg\forall x P(x) \vdash \exists x \neg P(x)$

Example: DeMorgan's laws 1 (solution)

$\neg\forall x P(x) \vdash \exists x \neg P(x)$

1	$\neg\forall x\, P(x)$	
2	$\quad \neg\exists x\, \neg P(x)$	
3	$\quad\quad c$	
4	$\quad\quad\quad \neg P(c)$	
5	$\quad\quad\quad \exists x\, \neg P(x)$	$\exists$I: 4
6	$\quad\quad\quad \bot$	$\bot$I: 5, 2
7	$\quad\quad \neg\neg P(c)$	$\neg$I: 4–6
8	$\quad\quad P(c)$	$\neg$E: 7
9	$\quad \forall x\, P(x)$	$\forall$I: 3–8
10	$\quad \bot$	$\bot$I: 9, 1
11	$\neg\neg\exists x\, \neg P(x)$	$\neg$I: 2–10
12	$\exists x\, \neg P(x)$	$\neg$E: 11

Example: DeMorgan's laws 2

$\neg\exists x\ P(x) \vdash \forall x\ \neg P(x)$

1	$\neg\exists x\ P(x)$	
2	$\quad c$	
3	$\quad\quad P(c)$	
4	$\quad\quad \exists x\ P(x)$	$\exists$I: 3
5	$\quad\quad \bot$	$\bot$I: 4, 1
6	$\quad \neg P(c)$	$\neg$I: 3–5
7	$\forall x\ \neg P(x)$	$\forall$I: 2–6

Example: DeMorgan's laws 2

$\neg \exists x \; P(x) \vdash \forall x \, \neg P(x)$

Example: DeMorgan's laws 2 (Solution)

$\neg\exists x\, P(x) \vdash \forall x\, \neg P(x)$

1	$\neg\exists x\, P(x)$	
2	$\quad c$	
3	$\quad\quad P(c)$	
4	$\quad\quad \exists x\, P(x)$	$\exists$I: 3
5	$\quad\quad \bot$	$\bot$I: 4, 1
6	$\quad \neg P(c)$	$\neg$I: 3–5
7	$\forall x\, \neg P(x)$	$\forall$I: 2–6