

Introduction to Logic *Relational Fitch Proofs*

Michael Genesereth
Computer Science Department
Stanford University

Welcome back. In previous lessons, we talked about the syntax and semantics of relational logic. We also looked at the notions of validity, contingency, and satisfiability, and the notions of logical equivalence, logical entailment, and consistency. Of these, logical entailment is the most interesting.

Logical Entailment

A set of premises Δ logically entails a conclusion φ ($\Delta \models \varphi$) if and only if every interpretation that satisfies Δ also satisfies φ .

Logical entailment for Relational Logic is defined the same as for Propositional Logic. A set of premises logically entails a conclusion if and only if every truth assignment that satisfies the premises also satisfies the conclusions.

Determining Logical Entailment

$$\{m \Rightarrow p \vee q, p \Rightarrow q\} \models m \Rightarrow q?$$

m	p	q	$m \Rightarrow p \vee q$	$p \Rightarrow q$	$m \Rightarrow q$
1	1	1	1	1	1
1	1	0	1	0	0
1	0	1	1	1	1
1	0	0	0	1	0
0	1	1	1	1	1
0	1	0	1	0	1
0	0	1	1	1	1
0	0	0	1	1	1

As in Propositional Logic, it is in principle possible to build a truth table for any set of sentences in Relational Logic. This truth table can then be used to determine validity, satisfiability, and so forth or to determine logical entailment and logical equivalence.

As an example, let us assume we have a language with just two object constants a and b and two relation constants p and q . Now consider the sentences shown below, and assume we want to know whether these sentences logically entail $\exists x.q(x)$.

$p(a) \vee p(b)$

$\forall x.(p(x) \Rightarrow q(x))$

A truth table for this problem is shown below. Each of the first four columns represents one

of the elements of the Herbrand base for this language. The two middle columns represent our premises, and the final column represents the conclusion.

Looking at the table, we see that there are 12 truth assignments that make the first premise true and nine that make the second premise true and five that make them both true (rows 1, 5, 6, 9, and 11). Note that every truth assignment that makes both premises true also makes the conclusion true. Hence, the premises logically entail the conclusion.

Determining Logical Entailment

$\{p(a) \vee p(b), \forall x.(p(x) \Rightarrow q(x))\} \models \exists x.q(x)?$

$p(a)$	$p(b)$	$q(a)$	$q(b)$	$p(a) \vee p(b)$	$\forall x.(p(x) \Rightarrow q(x))$	$\exists x.q(x)$
1	1	1	1	1	1	1
1	1	1	0	1	0	1
1	1	0	1	1	0	1
1	1	0	0	1	0	0
1	0	1	1	1	1	1
1	0	1	0	1	1	1
1	0	0	1	1	0	1
1	0	0	0	1	0	0
0	1	1	1	1	1	1
0	1	1	0	1	0	1
0	1	0	1	1	1	1
0	1	0	0	1	0	0
0	0	1	1	0	1	1
0	0	1	0	0	1	1
0	0	0	1	0	1	1
0	0	0	0	0	1	0

As in Propositional Logic, it is in principle possible to build a truth table for any set of sentences in Relational Logic. This truth table can then be used to determine validity, satisfiability, and so forth or to determine logical entailment and logical equivalence.

As an example, let us assume we have a language with just two object constants a and b and two relation constants p and q . Now consider the sentences shown below, and assume we want to know whether these sentences logically entail $\exists x.q(x)$.

$p(a) \vee p(b)$

$\forall x.(p(x) \Rightarrow q(x))$

A truth table for this problem is shown below. Each of the first four columns represents one of the elements of the Herbrand base for this language. The two middle columns represent our premises, and the final column represents the conclusion.

Looking at the table, we see that there are 12 truth assignments that make the first premise true and nine that make the second premise true and five that make them both true (rows 1, 5, 6, 9, and 11). Note that every truth assignment that makes both premises true also makes the conclusion true. Hence, the premises logically entail the conclusion.

Analysis

Object constants: n
Binary relation constants: k
Factoids in Herbrand Base: $k \cdot n^2$
Interpretations: $2^{k \cdot n^2}$

Object constants: 4
Binary relation constants: 4
Factoids in Herbrand Base: 64
Interpretations: $2^{64} = 18,446,744,073,709,551,616$

5

Good News

Good News: If Δ logically entails φ , then there is a finite proof of φ from Δ . And vice versa.

Good News: If Δ logically entails φ , it is possible to find such a proof in finite time.

More Good News: Such proofs are often *much* smaller than the corresponding truth tables.

All is not lost. As with Propositional Logic, we can define a proof system for Relational Logic. Interestingly, it is possible to show that a set of premises logically entails a conclusion if and only if the premises logically entail the conclusion. If there is a proof, it is possible to find such a proof in finite time. Moreover, such proofs are typically smaller than the corresponding truth tables.

Fitch System for Relational Logic

Logical Entailment and Provability

A set of premises Δ *logically entails* a conclusion ϕ ($\Delta \models \phi$) if and only if every interpretation that satisfies Δ also satisfies ϕ .

If there exists a proof of a sentence ϕ from a set Δ of premises using the rules of inference in R , we say that ϕ is *provable* from Δ using R (written $\Delta \vdash_R \phi$).

A reminder that, in talking about Logic, we have two notions - logical entailment and provability. A set of premises logically entails a conclusion if and only if every truth assignment that satisfies the premises also satisfies the conclusion. A sentence is provable from a set of premises if and only if there is a finite proof of the conclusion from the premises. The concepts are quite different. One is based on truth assignments; the other is based on symbolic manipulation of expressions.

Soundness and Completeness

A proof system is *sound* if and only if every provable conclusion is logically entailed.

If $\Delta \vdash \phi$, then $\Delta \models \phi$.

A proof system is *complete* if and only if every logical conclusion is provable.

If $\Delta \models \phi$, then $\Delta \vdash \phi$.

A proof system is sound if and only if every provable conclusion is logically entailed. In other words, if $\Delta \vdash \phi$, then $\Delta \models \phi$. We say that a proof system is complete if and only if every logical conclusion is provable. In other words, if $\Delta \models \phi$, then $\Delta \vdash \phi$.

Fitch

Theorem: Fitch is sound and complete for **Relational Logic**.

$\Delta \models \phi$ if and only if $\Delta \vdash_{\text{Fitch}} \phi$.

Upshot: The truth table method and the proof method succeed in exactly the same cases!

As with Propositional Logic, the Fitch system is sound and complete for Relational Logic. In other words, for this system, logical entailment and provability are identical. An arbitrary set of sentences Δ logically entails an arbitrary sentence ϕ if and only if ϕ is provable from Δ using Fitch.

Logical Rules of Inference

Negation Introduction
Negation Elimination

And Introduction
And Elimination

Or Introduction
Or Elimination

Assumption
Implication Elimination
Implication Introduction

Biconditional Introduction
Biconditional Elimination

In the Fitch system for Relational Logic, we include all of the rules of inference for Propositional Logic. We have introduction and elimination rules for conjunctions, disjunctions, implications, and biconditionals.

New Rules of Inference

Universal Elimination Domain Closure Universal Introduction	<i>Universal Reasoning</i>
Existential Introduction Existential Elimination	<i>Existential Reasoning</i>

To these rules of inference for logical operators, we add five new rules for quantified sentences - introduction and elimination rules for both universal and existential sentences and domain closure. Let's look at them in turn. If you're like me, the prospect of going through a discussion of five rules of inference sounds a little repetitive and boring. However, it is not so bad. Each of the rules has its own quirks and idiosyncrasies, its own personality. In fact, a couple of the rules suffer from a distinct excess of personality. If we are to use the rules correctly, we need to understand these idiosyncrasies. That's the goal of today's lecture.

Universal Elimination

Universal Elimination (UE)

$$\frac{\forall v.\phi}{\phi_{v=\tau}}$$

where τ is ground

NB: $\phi_{v=\tau}$ is an instance of ϕ with *all* occurrences of v replaced by τ .
For example $p(x,x)_{x=b}$ is $p(b,b)$.

Universal Elimination (UE) allows us to reason from the general to the particular. It states that, whenever we believe a universally quantified sentence, we can infer a version of the scope of that sentence in which the universally quantified variable is replaced by any ground term. (In the rule shown here, the notation $\phi_{v=\tau}$ means an instance of ϕ in which all occurrences of v have been replaced by τ .)

UE Examples

Premise:

$$\forall x.hates(jane,x)$$

Conclusions:

$$\begin{array}{ll} hates(jane,jill) & x \Leftarrow jill \\ hates(jane,jane) & x \Leftarrow jane \end{array}$$

Non-Conclusions:

$$hates(jane,y) \quad x \Leftarrow y \quad \text{Wrong!}$$

Must be ground.

Consider the sentence $\forall x.hates(jane,x)$. Jane hates everyone. From this premise, we can infer that Jane hates Jill, i.e. $hates(jane,jill)$. We can even infer that Jane hates herself, i.e. $hates(jane,jane)$.

Consider the sentence $\forall x.hates(x,x)$. Everyone hates herself. From this premise, we can infer that Jane hates Jill, i.e. $hates(jane,jill)$. We can even infer that Jane hates herself, i.e. $hates(jane,jane)$.

Note that we are not permitted to infer conclusions with free variables. For example, we cannot infer $hates(jane,x)$ or $hates(jane,y)$! No open sentences in Fitch.

UE Examples

Premise:

$$\forall x.hates(x,x)$$

Conclusions:

$$\begin{array}{ll} hates(jane,jane) & x \Leftarrow jane \\ hates(jill,jill) & x \Leftarrow jill \end{array}$$

Non-Conclusions:

$$\begin{array}{lll} hates(jane,x) & x \Leftarrow jane & \text{Wrong!} \\ hates(x,jane) & x \Leftarrow jane & \text{Wrong!} \\ hates(x,x) & x \Leftarrow jane & \text{Wrong!} \end{array}$$

Must be ground.

Consider the sentence $\forall x.hates(jane,x)$. Jane hates everyone. From this premise, we can infer that Jane hates Jill, i.e. $hates(jane,jill)$. We can even infer that Jane hates herself, i.e. $hates(jane,jane)$.

Consider the sentence $\forall x.hates(x,x)$. Everyone hates herself. From this premise, we can infer that Jane hates Jill, i.e. $hates(jane,jill)$. We can even infer that Jane hates herself, i.e. $hates(jane,jane)$.

In addition, we can use Universal Elimination to create conclusions with free variables. For example, we can infer $hates(jane,x)$ or, equivalently, $hates(jane,y)$.

UE Examples

Premise:

$\forall x. \exists y. \text{hates}(x, y)$

Conclusions:

$\exists y. \text{hates}(\text{jane}, y)$ $x \leftarrow \text{jane}$

UE Examples

Premise:

$\forall x. \forall y. \text{hates}(x, y)$

Conclusion:

$\forall y. \text{hates}(\text{jane}, y)$ $x \leftarrow \text{jane}$

Subsequent Conclusion:

$\text{hates}(\text{jane}, \text{jill})$ $y \leftarrow \text{jill}$
 $\text{hates}(\text{jane}, \text{jane})$ $y \leftarrow \text{jane}$

Domain Closure

Universal Introduction (UI) is the opposite of Universal Elimination - it allows us to reason from sentences about individuals to universally quantified versions of those sentences.

Unfortunately, it is not as easy as Universal Elimination. Just because we can prove that a sentence is true of one specific object does not mean that it is true of all objects. Of course, we could try proving a sentence about every known object.

Domain Closure

$$\frac{\begin{array}{c} \phi[\sigma_1] \\ \dots \\ \phi[\sigma_n] \end{array} \quad \left. \begin{array}{l} \text{every} \\ \text{object} \\ \text{constant} \end{array} \right\}}{\forall v. \phi[v]}$$

Domain closure is easy. For a language with object constants $\sigma_1, \dots, \sigma_n$, the rule is written as shown here. If we believe a schema is true for every instance, then we can infer a universally quantified version of that schema.

DC Example

$$\frac{\begin{array}{l} \text{likes}(\text{abby}, \text{cody}) \\ \text{likes}(\text{bess}, \text{cody}) \\ \text{likes}(\text{cody}, \text{cody}) \\ \text{likes}(\text{dana}, \text{cody}) \end{array}}{\forall x. \text{likes}(x, \text{cody})}$$

Suppose, for example, we have a language with just 4 object constants. Then we can write a domain closure rule as shown here. If a schema is true of abby and bess and cody and dana, then it is true of everyone.

Note that in applying Domain Closure, we do not need to worry about placeholders. Placeholders stand for constants in our language. If we have managed to prove that a condition holds for all of our object constants, then it must be true of everything, including whatever objects our placeholders represent.

Why restrict DC to languages with finitely many ground terms? Why not use domain closure rules for languages with infinitely many ground terms as well? It would be good if we could, but this would require rules of infinite length, and we do not allow infinitely large sentences in our language. We can get the effect of such sentences through the use of induction, which is discussed in a later chapter.

DC Example

$$\frac{\begin{array}{l} \text{likes}(\text{abby}, \text{abby}) \\ \text{likes}(\text{bess}, \text{bess}) \\ \text{likes}(\text{cody}, \text{cody}) \\ \text{likes}(\text{dana}, \text{dana}) \end{array}}{\forall x. \text{likes}(x, x)}$$

Suppose, for example, we have a language with just 4 object constants. Then we can write a domain closure rule as shown here. If a schema is true of abby and bess and cody and dana, then it is true of everyone.

Note that in applying Domain Closure, we do not need to worry about placeholders. Placeholders stand for constants in our language. If we have managed to prove that a condition holds for all of our object constants, then it must be true of everything, including whatever objects our placeholders represent.

Why restrict DC to languages with finitely many ground terms? Why not use domain closure rules for languages with infinitely many ground terms as well? It would be good if we could, but this would require rules of infinite length, and we do not allow infinitely large sentences in our language. We can get the effect of such sentences through the use of induction, which is discussed in a later chapter.

DC Example

$likes(abby, cody) \Rightarrow likes(cody, abby)$
 $likes(bess, cody) \Rightarrow likes(cody, bess)$
 $likes(cody, cody) \Rightarrow likes(cody, cody)$
 $likes(dana, cody) \Rightarrow likes(cody, dana)$

 $\forall x.(likes(x, cody) \Rightarrow likes(cody, x))$

Note that Domain Closure applies to sentences of any sort - logical sentences and quantified sentences, not just atomic sentences.

The main problem with domain closure is that it can require a lot of work in cases where the universe contains many objects. Fortunately, there is a technique that allows us to derive universally quantified sentences without enumerating all individual objects.

DC Example

$\exists y.likes(abby, y)$
 $\exists y.likes(bess, y)$
 $\exists y.likes(cody, y)$
 $\exists y.likes(dana, y)$

 $\forall x.\exists y.likes(x, y)$

Note that Domain Closure applies to sentences of any sort - logical sentences and quantified sentences, not just atomic sentences.

The main problem with domain closure is that it can require a lot of work in cases where the universe contains many objects. Fortunately, there is a technique that allows us to derive universally quantified sentences without enumerating all individual objects.

Universal Introduction

The downside of domain closure is that we need to enumerate the desired conclusion for every object. In some cases, it is possible to go from premises to the desired conclusion without enumerating. The corresponding rule of inference is called Universal Introduction.

Proof

$\forall y.(likes(cody,y) \Rightarrow happy(y))$
 $\forall y.likes(cody,y)$
...
 $likes(cody,abby)$
 $happy(abby)$
...

Proof

$\forall y.(likes(cody,y) \Rightarrow happy(y))$
 $\forall y.likes(cody,y)$
...
 $likes(cody,abby)$
 $happy(abby)$
...
 $likes(cody,dana)$
 $happy(dana)$
...
 $\forall y.happy(y)$

Proof

1.	$\forall y.(likes(cody,y) \Rightarrow happy(y))$	Premise
2.	$\forall y.likes(cody,y)$	Premise
3.	$likes(cody,abby) \Rightarrow happy(abby)$	UE: 1
4.	$likes(cody,abby)$	UE: 2
5.	$happy(abby)$	IE: 4, 3
6.	$likes(cody,bess) \Rightarrow happy(bess)$	UE: 1
7.	$likes(cody,bess)$	UE: 2
8.	$happy(bess)$	IE: 6, 7
9.	$likes(cody,cody) \Rightarrow happy(cody)$	UE: 1
10.	$likes(cody,cody)$	UE: 2
11.	$happy(cody)$	IE: 9, 10
12.	$likes(cody,dana) \Rightarrow happy(dana)$	UE: 1
13.	$likes(cody,dana)$	UE: 2
14.	$happy(dana)$	IE: 12, 13
15.	$\forall y.happy(y)$	DC: 5, 8, 11, 14

Example

- | | |
|---|----------|
| 1. $\forall y.(likes(cody,y) \Rightarrow happy(y))$ | Premise |
| 2. $\forall y.likes(cody,y)$ | Premise |
| 3. $likes(cody,c) \Rightarrow happy(c)$ | UE: 1 |
| 4. $likes(cody,c)$ | UE: 2 |
| 5. $happy(c)$ | IE: 4, 3 |
| 6. $\forall y.happy(y)$ | UI: 5 |

Reasoning About *Arbitrary Objects*

If we can prove a property about an *arbitrary object*, then it must be true of all objects.

Common type of mathematical reasoning:

*Let c be an **arbitrary** object.*

We can prove that a particular property is true of c .

Therefore, the property is true of everything.

Once we have this new placeholder, we can use it like any other placeholder to derive conclusions. If we have not made any assumptions about this object, whatever we derive must be true of all objects. So we can derive any such conclusion by a universally quantified sentence in which we have replaced the placeholder with a variable, provided that the placeholder does not occur in any active assumptions.

Placeholders

A *placeholder* is a new type of symbol that *stands for* an arbitrary object constant but *is not itself* an object constant. Spelled the same as object constants.

Placeholders must be disjoint from object constants.

Object Constants: *abby, bess, cody, dana*

Placeholder: *c*

Sometimes written in brackets: *[c]*

Placeholders are used only within the Fitch procedure, never used outside of the procedure.

The key to Universal Introduction is the use of a new type of constant, called a placeholder. A placeholder is effectively a stand-in for an actual object constant; we just do not know which constant it is. In writing placeholders, we use square brackets to differentiate placeholders from ordinary object constants. For example, *c* is a placeholder.

Universal Introduction (UI)

$$\frac{\phi}{\forall u. \phi_{\tau \leftarrow u}}$$

where τ is a placeholder
not used in any *active* assumption

NB: $\phi_{\tau \leftarrow v}$ is an instance of ϕ with *all* occurrences of τ replaced by v .

Universal Introduction (UI) allows us to reason from an arbitrary sentence with a placeholder to universally quantified versions of that sentence. Provided, of course, that we have not made any assumptions about that placeholder.

UI Example

Object Constants: *jane*, ...
Placeholders: *c*, ...

Premise:

hates(c,jane)

Conclusion:

$\forall x. \text{hates}(x, \text{jane})$

UI Example

Object Constants: *jane*, ...
Placeholders: *c*, ...

Premise:

$\text{hates}(c, \text{jane}) \Rightarrow \text{hates}(\text{jane}, c)$

Conclusion:

$\forall x. (\text{hates}(x, \text{jane}) \Rightarrow \text{hates}(\text{jane}, x)) \quad c \Leftarrow x$

Example

Premises:

$$\forall x.(p(x) \Rightarrow q(x))$$

$$\forall x.p(x)$$

Goal:

$$\forall x.q(x)$$

Let's look at a problem to see how we can use placeholders in conjunction with Universal Elimination and Universal Introduction to prove a simple conclusion.

Proof

- | | |
|--|----------|
| 1. $\forall x.(p(x) \Rightarrow q(x))$ | Premise |
| 2. $\forall x.p(x)$ | Premise |
| 3. $p(c) \Rightarrow q(c)$ | UE: 1 |
| 4. $p(c)$ | UE: 2 |
| 5. $q(c)$ | IE: 4, 3 |
| 6. $\forall x.q(x)$ | UI: 5 |

Problem

Premises:

$$\forall x.(p(x) \Rightarrow q(x))$$

$$\forall x.(q(x) \Rightarrow r(x))$$

Goal:

$$\forall x.(p(x) \Rightarrow r(x))$$

Let's look at a problem to see how we can use placeholders in conjunction with Universal Elimination and Universal Introduction to prove a simple conclusion.

Proof

1.	$\forall x.(p(x) \Rightarrow q(x))$	Premise
2.	$\forall x.(q(x) \Rightarrow r(x))$	Premise
3.	$p(c) \Rightarrow q(c)$	UE: 1
4.	$q(c) \Rightarrow r(c)$	UE: 2
5.	$p(c)$	Assumption
6.	$q(c)$	IE: 5, 3
7.	$r(c)$	IE: 6, 4
8.	$p(c) \Rightarrow r(c)$	II: 5, 7
9.	$\forall x.(p(x) \Rightarrow r(x))$	UI: 8

Lovers

Everybody loves somebody. Everybody loves a lover.
Show that everybody loves everybody.

Premises:

- $\forall y.\exists z.loves(y,z)$
- $\forall x.\forall y.(\exists z.loves(y,z) \Rightarrow loves(x,y))$

Conclusion:

- $\forall x.\forall y.loves(x,y)$

Now, let's consider a slightly more interesting version of this problem. We start with the same premises. However, our goal now is to prove the somewhat stronger result that everyone loves everyone. For all x and for all y, x loves y. Our premises in this case are the same as before, but our conclusion is different.

Proof

Everybody loves somebody. Everybody loves a lover.
Show that everybody loves everybody.

1.	$\forall y.\exists z.loves(y,z)$	Premise
2.	$\forall x.\forall y.(\exists z.loves(y,z) \Rightarrow loves(x,y))$	Premise
3.	$\exists z.loves(d,z)$	UE: 1
4.	$\forall y.(\exists z.loves(y,z) \Rightarrow loves(c,y))$	UE: 2
5.	$\exists z.loves(d,z) \Rightarrow loves(c,d)$	UE: 4
6.	$loves(c,d)$	IE: 5, 3
7.	$\forall y.loves(c,y)$	UI: 6
8.	$\forall x.\forall y.loves(x,y)$	UI: 7

The proof shown below is analogous to our previous proof. The only difference is that we have free variables in place of object constants at various points in the proof. Once again, our premises appear on lines 1 and 2. Once again, we use Universal Elimination to produce the result on line 3; but this time the result contains a free variable. We get the results on lines 4 and 5 by successive application of Universal Elimination to the sentence on line 2. We deduce the result on line 6 using Implication Elimination. Finally, we use two applications of Universal Introduction to generalize our result and produce the desired conclusion.

This second example illustrates the power of free variables. We can manipulate them as though we are talking about specific individuals (though each one could be any object); and, when we are done, we can universalize them to derive universally quantified conclusions.

Universal Introduction (UI)

$$\frac{\phi}{\forall v. \phi_{\tau \leftarrow v}}$$

where τ is a placeholder
not used in any *active* assumption

NB: $\phi_{\tau \leftarrow v}$ is an instance of ϕ with *all* occurrences of τ replaced by v .

Universal Introduction (UI) allows us to reason from an arbitrary sentence with a placeholder to universally quantified versions of that sentence. Provided,, fo course, that we have not made any assumptions about that placeholder.

Bad, Bad, Bad "Proof"

- | | | |
|--|------------|--------|
| 1. $\forall x.(p(x) \Rightarrow q(x))$ | Premise | |
| 2. $p(a)$ | Premise | |
| 3. $p(c) \Rightarrow q(c)$ | UE: 1 | |
| 4. $p(c)$ | Assumption | |
| 5. $q(c)$ | IE: 5, 3 | |
| 6. $\forall y.q(y)$ | UI: 5 | NO!!! |
| 7. $p(c) \Rightarrow \forall y.q(y)$ | II: 4, 6 | |
| 8. $\forall x.(p(x) \Rightarrow \forall y.q(y))$ | UI: 7 | |
| 9. $p(a) \Rightarrow \forall y.q(y)$ | UE: 8 | |
| 10. $\forall y.q(y)$ | IE: 9, 2 | Wrong. |

Universal Introduction (UI)

$$\frac{\phi}{\forall v. \phi_{\tau \leftarrow v}}$$

where τ is a placeholder
not used in any *active* assumption

NB: $\phi_{\tau \leftarrow v}$ is an instance of ϕ with *all* occurrences of τ replaced by v .

Universal Introduction (UI) allows us to reason from an arbitrary sentence with a placeholder to universally quantified versions of that sentence. Provided,, fo course, that we have not made any assumptions about that placeholder.

Name Conflict Not Cool

1. $\forall y.(likes(cody,y) \Rightarrow happy(y))$ Premise
2. $likes(cody,abby)$ Premise
3. $likes(cody,abby) \Rightarrow happy(abby)$ UE: 1
4. $happy(abby)$ IE: 3, 2
5. $\forall y.happy(y)$ UI: 4 **Wrong.**

Reasoning Tip for Universal Reasoning

If you have some universal sentences and you want to prove a universal sentence, use placeholders to eliminate the universals, prove a specific conclusion, then generalize.

Proof

$\forall x.(p(x) \Rightarrow q(x)), \forall x.(q(x) \Rightarrow r(x)) \models \forall x.(p(x) \Rightarrow r(x))$

1. $\forall x.(p(x) \Rightarrow q(x))$ Premise
2. $\forall x.(q(x) \Rightarrow r(x))$ Premise
3. $p(c) \Rightarrow q(c)$ UE: 1
4. $q(c) \Rightarrow r(c)$ UE: 2
5. $p(c)$ Assumption
6. $q(c)$ IE: 5, 3
7. $r(c)$ IE: 6, 4
8. $p(c) \Rightarrow r(c)$ II: 5, 7
9. $\forall x.(p(x) \Rightarrow r(x))$ UI: 8

Existential Introduction

Existential Introduction (EI)

$$\frac{\phi}{\exists v. \phi_{\tau \leftarrow v}}$$

where τ is a constant

NB: $\phi_{\tau \leftarrow v}$ is an instance of ϕ with *0 or more* occurrences of τ replaced by v .
For example, $p(b,b)_{b \leftarrow x}$ could be $p(x,b)$ or $p(b,x)$ or $p(x,x)$.

Existential Introduction (EI) is easy. If we believe a sentence involving a ground term τ , then we can infer an existentially quantified sentence in which occurrences of τ have been replaced by the existentially quantified variable. (In the rule here, the notation $\phi_{\tau \leftarrow v}$ means an instance of ϕ in which zero or more occurrences of v have been replaced by τ .)

Note that Existential Introduction applies only in cases where the term being replaced is ground. This restriction is necessary to avoid incorrect conclusions. Suppose, for example, we had the sentence $\forall x.\text{hates}(x,x)$. If we were to use EI to replace the second occurrence of x , we would get the conclusion $\exists y.\forall x.\text{hates}(x,y)$, i.e. there is someone whom everyone hates. The original sentence says that everyone hates himself whereas this incorrect conclusion states that there is a single person whom everyone hates.

EI Examples

Premise:
 $\text{hates}(\text{jill}, \text{jill})$

Conclusions:
 $\exists x.\text{hates}(x,x)$
 $\exists x.\text{hates}(\text{jill}, x)$
 $\exists x.\text{hates}(x, \text{jill})$

Two Applications:
 $\exists x.\exists y.\text{hates}(x,y)$

For example, from the sentence $\text{hates}(\text{jill}, \text{jill})$, we can infer that there is someone who hates himself, i.e. $\exists x.\text{hates}(x,x)$. We can also infer that there is someone Jill hates, i.e. $\exists y.\text{hates}(\text{jill}, y)$, and there is someone who hates Jill, i.e. $\exists x.\text{hates}(x, \text{jill})$. And, by two applications of Existential Introduction, we can infer that someone hates someone, i.e. $\exists x.\exists y.\text{hates}(x,y)$.

Note that, in Existential Introduction, it is important to avoid variables that appear in the sentence being quantified. Without this restriction, starting from $\text{Ex.hates}(\text{jane}, x)$, we might deduce $\text{Ex.Ex.hates}(x,x)$. It is an odd sentence since it contains nested quantifiers of the same variable. However, it is a legal sentence, and it states that there is someone who hates himself, which does not follow from the premise in this case.

EI Examples

Premise:

$\forall x.hates(x,x)$

Non-Conclusion:

$\exists y.\forall x.hates(x,y)$

Wrong. Constants only!

Note that Existential Introduction applies only in cases where the term being replaced is ground. This restriction is necessary to avoid incorrect conclusions. Suppose, for example, we had the sentence $\forall x.hates(x,x)$. If we were to use EI to replace the second occurrence of x , we would get the conclusion $\exists y.\forall x.hates(x,y)$, i.e. there is someone whom everyone hates. The original sentence says that everyone hates himself whereas this incorrect conclusion states that there is a single person whom everyone hates.

Existential Elimination

EE Example

Premises:

$\exists x.hates(jane,x)$

$\exists x.hates(jane,x) \Rightarrow mean(jane)$

Conclusion:

$mean(jane)$

For example, if we have the sentence $\forall x.(hates(jane,x) \Rightarrow mean(jane))$ and we have the sentence $\exists x.hates(jane,x)$, then we can conclude $mean(jane)$.

EE Example

Metatheorem: $\forall v.(\phi \Rightarrow \psi)$ is equivalent to $(\exists v.\phi \Rightarrow \psi)$ so long as ψ is free of v .

Example:

$\exists x.hates(jane,x) \Rightarrow mean(jane)$
is equivalent to
 $\forall x.(hates(jane,x) \Rightarrow mean(jane))$

Proof Reminder:

$\forall v.(\phi \Rightarrow \psi)$ is equivalent to $\forall v.(\neg\phi \vee \psi)$,
which is equivalent to $(\forall v.\neg\phi \vee \psi)$ since ψ free of v ,
which is equivalent to $(\neg\exists v.\phi \vee \psi)$,
which is equivalent to $(\exists v.\phi \Rightarrow \psi)$.

For example, if we have the sentence $\forall x.(hates(jane,x) \Rightarrow mean(jane))$ and we have the sentence $\exists x.hates(jane,x)$, then we can conclude $mean(jane)$.

EE Example

Premises:

$\exists x.hates(jane,x)$
 $\forall x.hates(jane,x) \Rightarrow mean(jane)$

Equivalent Premises:

$\exists x.hates(jane,x)$
 $\forall x.(hates(jane,x) \Rightarrow mean(jane))$

Conclusion (by Implication Elimination):
 $mean(jane)$

For example, if we have the sentence $\forall x.(hates(jane,x) \Rightarrow mean(jane))$ and we have the sentence $\exists x.hates(jane,x)$, then we can conclude $mean(jane)$.

Existential Elimination (EE)

$\exists v.\phi$
 $\forall v.(\phi \Rightarrow \psi)$
—

ψ
where v does not occur free in ψ

Existential Elimination (EE) works in the opposite direction. Suppose we have an existentially quantified sentence with target ϕ ; and suppose we have a universally quantified implication in which the antecedent is the same as the scope of our existentially quantified sentence and the conclusion does not contain any occurrences of the quantified variable. Then, we can use Existential Elimination to infer the consequent.

EE Example

Premises:

$\exists x.hates(jane, x)$

$\forall x.(hates(jane, x) \Rightarrow mean(jane))$

Conclusion:

$mean(jane)$

For example, if we have the sentence $\forall x.(hates(jane, x) \Rightarrow mean(jane))$ and we have the sentence $\exists x.hates(jane, x)$, then we can conclude $mean(jane)$.

Existential Elimination (EE)

$\exists v.\phi$

$\forall v.(\phi \Rightarrow \psi)$

—————
 ψ

where v does not occur *free* in ψ

Existential Elimination (EE) works in the opposite direction. Suppose we have an existentially quantified sentence with target ϕ ; and suppose we have a universally quantified implication in which the antecedent is the same as the scope of our existentially quantified sentence and the conclusion does not contain any occurrences of the quantified variable. Then, we can use Existential Elimination to infer the consequent.

EE Example

Premises:

$\exists x.hates(jane, x)$

$\forall x.(hates(jane, x) \Rightarrow \forall x.hates(x, jane))$

Conclusion:

$\forall x.hates(x, jane)$

For example, if we have the sentence $\forall x.(hates(jane, x) \Rightarrow mean(jane))$ and we have the sentence $\exists x.hates(jane, x)$, then we can conclude $mean(jane)$.

Or Elimination

$$\frac{\begin{array}{l} \phi \vee \psi \\ \phi \Rightarrow \chi \\ \psi \Rightarrow \chi \end{array}}{\chi}$$

Or Introduction allows us to infer an arbitrary disjunction so long as at least one of the disjuncts is already in the proof. Or Elimination is a little more complicated than And Elimination. Since we do not know which of the disjuncts is true, we cannot just drop the $\vee$. However, if we know that every disjunct entails some sentence, then we can infer that sentence even if we do not know which disjunct is true.

EE = OE on Steroids

$$\begin{array}{ccc} \text{EE} & \frac{\begin{array}{l} \exists v.\phi \\ \forall v.(\phi \Rightarrow \psi) \end{array}}{\psi} & \xrightarrow{\quad} \frac{\begin{array}{l} \phi_1 \vee \dots \vee \phi_n \\ (\phi_1 \Rightarrow \psi) \wedge \dots \wedge (\phi_n \Rightarrow \psi) \end{array}}{\psi} \\ & & \updownarrow \\ & & \begin{array}{l} \phi_1 \vee \dots \vee \phi_n \\ \phi_1 \Rightarrow \psi \\ \dots \\ \phi_n \Rightarrow \psi \end{array} \\ \text{OE} & \frac{\begin{array}{l} \phi \vee \psi \\ \phi \Rightarrow \chi \\ \psi \Rightarrow \chi \end{array}}{\chi} & \xleftarrow{\quad} \frac{\quad}{\psi} \end{array}$$

Or Introduction allows us to infer an arbitrary disjunction so long as at least one of the disjuncts is already in the proof. Or Elimination is a little more complicated than And Elimination. Since we do not know which of the disjuncts is true, we cannot just drop the $\vee$. However, if we know that every disjunct entails some sentence, then we can infer that sentence even if we do not know which disjunct is true.

Intuition Analogous to Universal Reasoning

Suppose we know $\exists v.\phi(v)$.
 We hypothesize an object c and assume $\phi(c)$.
 We try to prove ψ .
 If ψ does not contain c , then it is true for *any* such c .
 We know that *there is some* v from $\exists v.\phi(v)$.
 So we can conclude ψ .

But we are still in the subproof.

Once we have this new placeholder, we can use it like any other placeholder to derive conclusions. If we have not made any assumptions about this object, whatever we derive must be true of all objects. So we can derive any such conclusion by a universally quantified sentence in which we have replaced the placeholder with a variable, provided that the placeholder does not occur in any active assumptions.

Intuition Analogous to Universal Reasoning

Suppose we know $\exists v.\phi(v)$.

We hypothesize an object c and assume $\phi(c)$.

We try to prove ψ .

If ϕ does not contain c , then it is true for *any* such c .

We know that *there is some* v from $\exists v.\phi(v)$.

So we can conclude ψ .

But we are still in the subproof.

So, we exit the subproof with $(\phi(c) \Rightarrow \psi)$.

We apply Universal Introduction to get $\forall v.(\phi(v) \Rightarrow \psi)$

Then we apply EE to get ψ outside the subproof.

Once we have this new placeholder, we can use it like any other placeholder to derive conclusions. If we have not made any assumptions about this object, whatever we derive must be true of all objects. So we can derive any such conclusion by a universally quantified sentence in which we have replaced the placeholder with a variable, provided that the placeholder does not occur in any active assumptions.

Reasoning Tip for Existential Reasoning

If you have an existential sentence,

- (1) assume the scope with a placeholder instead of variable,
- (2) prove some conclusion,
- (3) exit the assumption with an implication,
- (4) generalize with Universal Introduction, and
- (5) use Existential Elimination to derive the conclusion.

Proof

$$\exists y.\forall x.likes(x,y) \models \forall x.\exists y.likes(x,y)$$

1.	$\exists y.\forall x.likes(x,y)$	Premise
(1) 2.	$\forall x.likes(x,d)$	Assumption
3.	$likes(c,d)$	UE: 2
(2) 4.	$\exists y.likes(c,y)$	EI: 3
(3) 5.	$\forall x.likes(x,d) \Rightarrow \exists y.likes(c,y)$	II: 2, 4
(4) 6.	$\forall y.(\forall x.likes(x,y) \Rightarrow \exists y.likes(c,y))$	UI: 5
(5) 7.	$\exists y.likes(c,y)$	EE: 1, 6
8.	$\forall x.\exists y.likes(x,y)$	UI: 7

Useful Result

- | | | |
|----|-------------------------------------|------------|
| 1. | $\forall x.(p(x) \Rightarrow q(a))$ | Premise |
| 2. | $\exists x.p(x)$ | Assumption |
| 3. | $q(a)$ | EE: 2, 1 |
| 4. | $\exists x.p(x) \Rightarrow q(a)$ | II: 2, 3 |

Another Useful Result

- | | | |
|----|-------------------------------------|------------|
| 1. | $\exists x.p(x) \Rightarrow q(a)$ | Premise |
| 2. | $p(c)$ | Assumption |
| 3. | $\exists x.p(x)$ | EI: 2 |
| 4. | $q(a)$ | IE: 1, 3 |
| 5. | $p(c) \Rightarrow q(a)$ | II: 2, 4 |
| 6. | $\forall x.(p(x) \Rightarrow q(a))$ | UI: 5 |

Fitch Online System

Course Website

<http://logica.stanford.edu>

Proof

- | | |
|--|----------|
| 1. $\forall x.(p(x) \Rightarrow q(x))$ | Premise |
| 2. $\forall x.p(x)$ | Premise |
| 3. $p(c) \Rightarrow q(c)$ | UE: 1 |
| 4. $p(c)$ | UE: 2 |
| 5. $q(c)$ | IE: 4, 3 |
| 6. $\forall x.q(x)$ | UI: 5 |

Proof

$\forall x.(p(x) \Rightarrow q(x)), \forall x.(q(x) \Rightarrow r(x)) \models \forall x.(p(x) \Rightarrow r(x))$

- | | |
|--|------------|
| 1. $\forall x.(p(x) \Rightarrow q(x))$ | Premise |
| 2. $\forall x.(q(x) \Rightarrow r(x))$ | Premise |
| 3. $p(c) \Rightarrow q(c)$ | UE: 1 |
| 4. $q(c) \Rightarrow r(c)$ | UE: 2 |
| 5. $p(c)$ | Assumption |
| 6. $q(c)$ | IE: 5, 3 |
| 7. $r(c)$ | IE: 6, 4 |
| 8. $p(c) \Rightarrow r(c)$ | II: 5, 7 |
| 9. $\forall x.(p(x) \Rightarrow r(x))$ | UI: 8 |

Fitch

UndoCopyPasteLoadSaveLibraryHelp

+

-

Objects:abby,bess,cody,dana

+

-

Functions:

Select All

1.

likes(abby,bess)Premise

2.

likes(bess,cody)Premise

3.

likes(cody,dana)Premise

4.

likes(dana,dana)Premise

5.

EYlikes(abby,Y)Existential Introduction: 1

6.

EYlikes(bess,Y)Existential Introduction: 2

7.

EYlikes(cody,Y)Existential Introduction: 3

8.

EYlikes(dana,Y)Existential Introduction: 4

9.

AXEYlikes(X,Y)Domain Closure: 5, 6, 7, 8

Premise

Negation Introduction

Assumption

Universal Introduction

Domain Closure

Reiteration

Negation Elimination

Implication Introduction

Universal Elimination

Induction

Truthable

And introduction

Implication Elimination

Existential Introduction

Equality Introduction

Shortcut

And Elimination

Biconditional Introduction

Existential Elimination

Equality Elimination

Replace

Or introduction

Biconditional Elimination

Rational Equation

Contradict

Or Elimination

Rational Induction

Delete

An aerial photograph of a large, historic building complex, likely a university campus. The building features multiple wings with red-tiled roofs and light-colored walls. A central courtyard with a green lawn is visible. The surrounding area is lush with greenery, including palm trees and other tropical plants. The image is framed by a thin black border.