CHAPTER 7

Relational Proofs

7.1 Introduction

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. In the case of Propositional Logic, we can check logical entailment by examining a truth table for the language. With finitely many proposition constants, the truth truth is large but finite. For Relational Logic, things are not so easy. It is possible to have Herbrand bases of infinite size; and, in such cases, truth assignments are infinitely large and there are infinitely many of them, making it impossible to check logical entailment using truth tables.

All is not lost. As with Propositional Logic, we can establish logical entailment in Relational Logic by writing proofs. In fact, it is possible to show that, with a few simple restrictions, a set of premises logically entails a conclusions if and only if there is a finite proof of the conclusion from the premises, even when the Herbrand base is infinite. Moreover, it is possible to find such proofs in a finite time. That said, things are not perfect. If a set of sentences does *not* logically entail a conclusion, then the process of searching for a proof might go on forever. Moreover, if we remove the restrictions mentioned above, we lose the guarantee of finite proofs. Still, the relationship between logical entailment and finite provability, given those restrictions, is a very powerful result and has enormous practical benefits.

In this chapter, we start by extending the Fitch system from Propositional Logic to Relational Logic. We then illustrate the system with a few examples. Finally, we talk about soundness and completeness.

7.2 Proofs

Formal proofs in Relational Logic are analogous to formal proofs in Propositional Logic. The major difference is that there are additional mechanisms to deal with identities and quantified sentences.

The Fitch system for Relational Logic is an extension of the Fitch system for Propositional Logic. In addition to the ten logical rules of inference, there are four ordinary rules of inference for quantified sentences and one additional rule for finite languages. Let's look at each of these in turn. (If youÕre like me, the prospect of going through a discussion of so many 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.)

Universal Introduction (UI) allows us to reason from arbitrary sentences to universally quantified versions of those sentences.

Universal Introduction

$$\frac{\phi}{\forall\nu.\phi}$$

where ν does not occur free in both ϕ and an active assumption

Typically, UI is used on sentences with free variables to make their quantification explicit. For example, if we have the sentence $hates(jane,y)$, then, we can infer $\forall y.hates(jane,y)$.

Note that we can also apply the rule to sentences that do not contain the variable that is quantified in the conclusion. For example, from the sentence $hates(jane,jill)$, we can infer $\forall x.hates(jane,jill)$. And, from the sentence $hates(jane,y)$, we can infer $\forall x.hates(jane,y)$. These are not particularly sensible conclusions. However, the results are correct, and the deduction of such results is necessary to ensure that our proof system is complete.

There is one important restriction on the use of Universal Introduction. If the variable being quantified appears in the sentence being quantified, it must not appear free in any *active assumption*, i.e. an assumption in the current subproof or any superproof of that subproof. For example, if there is a subproof with assumption $p(x)$ and from that we have managed to derive $q(x)$, then we cannot just write $\forall x.q(x)$.

If we want to quantify a sentence in this situation, we must first use Implication Introduction to discharge the assumption and then we can apply Universal Introduction. For example, in the case just described, we can first apply Implication Introduction to derive the result $(p(x) \Rightarrow q(x))$ in the parent of the subproof containing our assumption, and we can then apply Universal Introduction to derive $\forall x.(p(x) \Rightarrow q(x))$.

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 target of that sentence in which the universally quantified variable is replaced by an appropriate term.

Universal Elimination

$$\frac{\forall\nu.\phi[\nu]}{\phi[\tau]}$$

where τ is substitutable for ν in ϕ

For example, consider the sentence $\forall y.hates(jane,y)$. From this premise, we can infer that Jane hates Jill, i.e. $hates(jane,jill)$. We also can infer that Jane hates her mother, i.e. $hates(jane,mother(jane))$. We can even infer than Jane hates herself, i.e. $hates(jane,jane)$.

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

In using Universal Elimination, we have to be careful to avoid conflicts with other variables and quantifiers in the quantified sentence. This is the reason for the constraint on the replacement term. As an example of what can go wrong without this constraint, consider the sentence $\forall x.\exists y.hates(x,y)$, i.e. everybody hates somebody. From this sentence, it makes sense to infer $\exists y.hates(jane,y)$, i.e. Jane hates somebody. However, we do not want to infer $\exists y.hates(y,y)$; i.e., there is someone who hates herself.

We can avoid this problem by obeying the restriction on the Universal Elimination rule. We say that a term τ is *free* for a variable ν in a sentence ϕ if and only if no free occurrence of ν occurs within the scope of a quantifier of some variable in τ. For example, the term x is free for y in $\exists z.hates(y,z)$. However, the term z is not free for y, since y is being replaced by z and y occurs within the scope of a quantifier of z. Thus, we cannot substitute z for y in this sentence, and we avoid the problem we have just described.

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 one, some, or all occurrences of τ have been replaced by the existentially quantified variable.

Existential Introduction

$$\begin{array}{l} \phi[\tau] \\ \hline \exists\nu.\phi[\nu] \end{array}$$

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

Note that, in Existential Introduction, it is important to avoid variables that appear in the sentence being quantified. Without this restriction, starting from $\exists x.hates(jane,x)$, we might deduce $\exists x.\exists x.hates(x,x)$. 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.

Existential Elimination (EE). 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.

Existential Elimination

$$\begin{array}{l} \exists\nu.\phi[\nu] \\ \forall\nu.(\phi[\nu] \Rightarrow \psi) \\ \hline \psi \\ \text{where } \nu \text{ does not occur in } \psi \end{array}$$

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

It is interesting to note that Existential Elimination is analogous to Or Elimination. This is as it should, as an existential sentence is effectively a disjunction. Recall that, in Or Elimination, we start with a disjunction with n disjuncts and n implications, one for each of the disjuncts and produce as conclusion the consequent shared by all of the implications. An existential sentence (like the first premise in any instance of Existential Elimination) is effectively a disjunction over the set of all ground terms; and a universal implication (like the second premise in any instance of Existential Elimination) is effectively a set of implications, one for each ground term in the language. The conclusion of Existential Elimination is the common consequent of these implications, just as in Or Elimination.

Finally, for languages with finite Herbrand bases, we have the *Domain Closure* (DC) rule. For a language with object constants $\sigma_1, \ldots, \sigma_n$, the rule is written as shown below. If we believe a schema is true for every instance, then we can infer a universally quantified version of that schema.

$$\begin{array}{l} \textbf{Domain Closure} \\ \phi[\sigma_1] \\ \quad \ldots \\ \phi[\sigma_n] \\ \hline \forall \nu.\phi[\nu] \end{array}$$

For example, in a language with four object constants *a* and *b* and *c* and *d*, we can derive the conclusion $\forall x.\phi[x]$ whenever we have $\phi[a]$ and $\phi[b]$ and $\phi[c]$ and $\phi[d]$.

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.

As in Chapter 3, we define a *structured proof* of a conclusion from a set of premises to be a sequence of (possibly nested) sentences terminating in an occurrence of the conclusion at the *top level* of the proof. Each step in the proof must be either (1) a premise (at the top level) or an assumption (other than at the top level) or (2) the result of applying an ordinary or structured rule of inference to earlier items in the sequence (subject to the constraints given above and in Chapter 3).

7.3 Example

As an illustration of these concepts, consider the following problem. Suppose we believe that everybody loves somebody. And suppose we believe that everyone loves a lover. Our job is to prove that Jack loves Jill.

First, we need to formalize our premises. Everybody loves somebody. For all *y*, there exists a *z* such that *loves*(*y*,*z*).

$$\forall y.\exists z.loves(y,z)$$

Everybody loves a lover. If a person is a lover, then everyone loves him. If a person loves another person, then everyone loves him. For all *x* and for all *y* and for all *z*, *loves*(*y*,*z*) implies *loves*(*x*,*y*).

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

Our goal is to show that Jack loves Jill. In other words, starting with the preceding sentences, we want to derive the following sentence.

$$loves(jack,jill)$$

A proof of this result is shown below. Our premises appear on lines 1 and 2. The sentence on line 3 is the result of applying Universal Elimination to the sentence on line 1, substituting *jill* for y. The sentence on line 4 is the result of applying Universal Elimination to the sentence on line 2, substituting *jack* for x. The sentence on line 5 is the result of applying Universal Elimination to the sentence on line 4, substituting *jill* for y. Finally, we apply Existential Elimination to produce our conclusion on line 6.

1.	$\forall y.\exists z.loves(y,z)$	Premise
2.	$\forall x.\forall y.\forall z.(loves(y,z) \Rightarrow loves(x,y))$	Premise
3.	$\exists z.loves(jill,z)$	UE: 1
4.	$\forall y.\forall z.(loves(y,z) \Rightarrow loves(jack,y))$	UE: 2
5.	$\forall z.(loves(jill,z) \Rightarrow loves(jack,jill))$	UE: 4
6.	$loves(jack,jill)$	EE: 3, 5

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.

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

The proof shown below is analogous to the proof above. 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. Note that we have replaced the 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 Existential Elimination. Finally, we use two applications of Universal Introduction to generalize our result and produce the desired conclusion.

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

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.

7.4 Example

As another illustration of Relational Logic proofs, consider the following problem. We know that horses are faster than dogs and that there is a greyhound that is faster than every rabbit. We know that Harry is a horse and that Ralph is a rabbit. Our job is to derive the fact that Harry is faster than Ralph.

First, we need to formalize our premises. The relevant sentences follow. Note that we have added two facts about the world not stated explicitly in the problem: that greyhounds are dogs and that our *faster than* relationship is transitive.

$$\begin{array}{l} \forall x.\forall y.(h(x) \wedge d(y) \Rightarrow f(x,y)) \\ \exists y.(g(y) \wedge \forall z.(r(z) \Rightarrow f(y,z))) \\ \forall y.(g(y) \Rightarrow d(y)) \\ \forall x.\forall y.\forall z.(f(x,y) \wedge f(y,z) \Rightarrow f(x,z)) \\ h(harry) \\ r(ralph) \end{array}$$

Our goal is to show that Harry is faster than Ralph. In other words, starting with the preceding sentences, we want to derive the following sentence.

$$f(harry,ralph)$$

The derivation of this conclusion goes as shown below. The first six lines correspond to the premises just formalized. On line 7, we start a subproof with an assumption corresponding to the scope of the existential on line 2, with the idea of using Existential Elimination later on in the proof. Lines 8 and 9 come from And Elimination. Line 10 is the result of applying Universal Elimination to the sentence on line 9. On line 11, we use Implication Elimination to infer that *y* is faster than Ralph. Next, we instantiate the sentence about greyhounds and dogs and infer that *y* is a dog. Then, we instantiate the sentence about horses and dogs; we use And Introduction to form a conjunction matching the antecedent of this instantiated sentence; and we infer that Harry is faster than *y*. We then instantiate the transitivity sentence, again form the necessary conjunction, and infer the desired conclusion. Finally, we use Implication Introduction to discharge our subproof; we use Universal Introduction to universalize the results; and we use Existential Elimination to produce our desired conclusion.

1.	$\forall x.\forall y.(h(x) \wedge d(y) \Rightarrow f(x,y))$	Premise
2.	$\exists y.(g(y) \wedge \forall z.(r(z) \Rightarrow f(y,z)))$	Premise
3.	$\forall y.(g(y) \Rightarrow d(y))$	Premise
4.	$\forall x.\forall y.\forall z.(f(x,y) \wedge f(y,z) \Rightarrow f(x,z))$	Premise
5.	$h(harry)$	Premise
6.	$r(ralph)$	Premise
7.	$\quad\vert\; g(y) \wedge \forall z.(r(z) \Rightarrow f(y,z))$	Assumption
8.	$\quad\vert\; g(y)$	AE: 7
9.	$\quad\vert\; \forall z.(r(z) \Rightarrow f(y,z))$	AE: 7
10.	$\quad\vert\; r(ralph) \Rightarrow f(y,ralph)$	UE: 9
11.	$\quad\vert\; f(y,ralph)$	IE: 10, 6

12.	$g(y) \Rightarrow d(y)$	UE: 3
13.	$d(y)$	IE: 12, 8
14.	$\forall y.(h(harry) \wedge d(y) \Rightarrow f(harry,y))$	UE: 1
15.	$h(harry) \wedge d(y) \Rightarrow f(harry,y)$	UE: 14
16.	$h(harry) \wedge d(y)$	AI: 5, 13
17.	$f(harry,y)$	IE: 15, 16
18.	$\forall y.\forall z.(f(harry,y) \wedge f(y,z) \Rightarrow f(harry,z))$	UE: 4
19.	$\forall z.(f(harry,y) \wedge f(y,z) \Rightarrow f(harry,z))$	UE: 18
20.	$f(harry,y) \wedge f(y,ralph) \Rightarrow f(harry,ralph)$	UE: 19
21.	$f(harry,y) \wedge f(y,ralph)$	AI: 17, 11
22.	$f(harry,ralph)$	IE: 20, 21
23.	$g(y) \wedge \forall z.(r(z) \Rightarrow f(y,z)) \Rightarrow f(harry,ralph)$	II: 7, 22
24.	$\forall y.(g(y) \wedge \forall z.(r(z) \Rightarrow f(y,z)) \Rightarrow f(harry,ralph))$	UI: 23
25.	$f(harry,ralph)$	EE: 2, 24

This derivation is somewhat lengthy, but it is completely mechanical. Each conclusion follows from previous conclusions by a mechanical application of a rule of inference. On the other hand, in producing this derivation, we rejected numerous alternative inferences. Making these choices intelligently is one of the key problems in the process of inference.

7.5 Example

In this section, we use our proof system to prove some basic results involving quantifiers.

Given $\forall x.\forall y.(p(x,y) \Rightarrow q(x))$, we know that $\forall x.(\exists y.p(x,y) \Rightarrow q(x))$. In general, given a universally quantified implication, it is okay to drop a universal quantifier of a variable outside the implication and apply an existential quantifier of that variable to the antecedent of the implication, provided that the variable does not occur in the consequent of the implication.

Our proof is shown below. As usual, we start with our premise. We use Universal Elimination to strip away the quantifier. We then start a subproof with a conjunction as assumption. We use And Elimination to expose the individual conjuncts. This allows us to derive $q(x)$ using Implication Elimination. We then use Identity Elimination on our identity and this result to change the variable x to the variable y. Finally, we create an implication with Implication Introduction, and we generalize using two applications of Universal Introduction.

1.	$\forall x.\forall y.(p(x,y) \Rightarrow q(x))$	Premise
2.	$\exists y.p(x,y)$	Assumption
3.	$\forall y.(p(x,y) \Rightarrow q(x))$	UE: 1

4.	$\mid\ q(x)$	EE: 2, 3
5.	$\exists y.p(x,y) \Rightarrow q(x)$	II: 4
6.	$\forall x.(\exists y.p(x,y) \Rightarrow q(x))$	UI: 5

The relationship holds the other way around as well. Given $\forall x.(\exists y.p(x,y) \Rightarrow q(x))$, we know that $\forall x.\forall y.(p(x,y) \Rightarrow q(x))$. We can convert an existential quantifier in the antecedent of an implication into a universal quantifier outside the implication.

Our proof is shown below. As usual, we start with our premise. We use Universal Elimination to strip away the quantifier. We then start a subproof with a conjunction as assumption. We use And Elimination to expose the individual conjuncts. This allows us to derive $q(x)$ using Implication Elimination. We then use Identity Elimination on our identity and this result to change the variable x to the variable y. Finally, we create an implication with Implication Introduction, and we generalize using two applications of Universal Introduction.

1.	$\forall x.(\exists y.p(x,y) \Rightarrow q(x))$	Premise
2.	$\mid\ p(x,y)$	Assumption
3.	$\mid\ \exists y.p(x,y)$	EI: 2
4.	$\mid\ \exists y.p(x,y) \Rightarrow q(x)$	UE: 1
5.	$\mid\ q(x)$	IE: 4, 3
6.	$p(x,y) \Rightarrow q(x)$	II: 5
7.	$\forall x.\forall y.(p(x,y) \Rightarrow q(x))$	2 x UI: 6