Section 5.3 Entailment in predicate logic

The fundamental problem of quantifier logic is to decide if certain formulas *entail* another formula or not:

$$? \qquad \Phi_1, \Phi_2, \ldots \Phi_\ell \vdash \Psi . \qquad (1)$$

We say that (1) is the case, $\Phi_1, \Phi_2, \ldots \Phi_\ell$ *entail* Ψ, if the following holds: no matter how we specify a (non-empty) universe, and interpretations of the relation-symbols in all of the formulas $\Phi_1, \Phi_2, \ldots \Phi_\ell, \Psi$, and furthermore, no matter how we give values to all the free variables in all the formulas involved, every time when all the *premisses* $\Phi_1, \Phi_2, \ldots \Phi_\ell$ evaluate to $\top$, the *conclusion* Ψ also evaluates to $\top$.

Let us note that, in (1), the possibility $\ell=0$ is allowed. This means that Ψ is *identically true*, without any premiss, no matter what interpretation, and what values of the free variables in Ψ (if any) we take. We say that Ψ is *logically valid*, and write $\vdash \Psi$, if this is the case.

Of course, there are two possibilities with given premisses $\Phi_1, \Phi_2, \ldots \Phi_\ell$ and a given conclusion Ψ. Either (1) does hold, or it does not. The basic question is how we recognize if one or the other is the case.

To recognize that (1) does not hold,

$$? \qquad \Phi_1, \Phi_2, \ldots, \Phi_\ell \nvdash \Psi ,$$

what we need is an example (*counter-example*) consisting of an interpretation, and values of the free variables that make *all the premisses* **true** and the *conclusion* **false**.

For instance,

$$\forall x \exists y Rxy \nvdash \exists y \forall x Rxy , \qquad (2)$$

as the following very simple example shows:

$$U = \{1, 2\}$$
$$R = \{(1, 2), (2, 1)\} .$$

Now, $\forall x\exists yRxy \sim \top$: for every $x\in U$, there is $y\in U$ such that xRy : for $x=1$, we can take $y=2$, and for $x=2$, we can take $y=1$; but *it is not the case* that there is a $y\in U$ such that for all x, xRy : if $y=1$, then *not*-$1Ry$, and if $y=2$, then *not*-$2Ry$.

Another example showing (2) uses familiar material: we let $U=\mathbb{N}$, and $Rxy \sim x<y$. Then, clearly, $\forall x\exists yRxy \sim \top$, and $\exists y\forall xRxy \sim \bot$.

It must be said that it could be *very difficult* to find a counter-example to an entailment, even if such exists.

On the other hand, we have

$$\exists y\forall xRxy \vdash \forall x\exists yRxy .$$

This can be shown by a direct "logical argument", a simple mathematical proof, as follows.

We argue in any fixed, but otherwise unspecified, interpretation. Suppose that $\exists y\forall xRxy \sim \top$. Then there is $y\in U$ such that $\forall xR(x, y) \sim \top$. Let us fix this y . Now, to show that $\forall x\exists yRxy \sim \top$, let $x\in U$ be arbitrary. Using y specified above, we have $R(x, y) \sim \top$. This means that $\exists yR(x, y) \sim \top$. Since x was arbitrary in U, we have that $\forall x\exists yRxy \sim \top$.

This proof was very easy; in fact, it was trivial. However, once again, it could be *very difficult* to prove that a given entailment holds true, even if it is in fact true.

There is a systematic way of showing of an entailment is true -- if it *is* true. In fact, if an entailment holds true, we can *deduce* it in a specific way. We will explain how this is done using the framework of Boolean algebras.

First of all, note that the fact that there are more than one premiss on the left of (1) is not really essential; (1) is equivalent to

$$\Phi_1\wedge\Phi_2\wedge...\wedge\Phi_\ell \vdash \Psi ,$$

where we took the conjunction of the premisses to be the new, single, premiss. Also, $\vdash\Psi$ iff $\top\vdash\Psi$ (why?). Let us fix a certain set of relation symbols and variables, and consider the set of all formulas using these only; let $\mathcal{F}$ be the set of all formulas. Therefore, what we really have is a *binary relation* $\vdash$ *called entailment on the set* $\mathcal{F}$. Now, we observe, very directly, that the relation $\vdash$ is reflexive, and transitive:

$\Phi\vdash\Phi$;
$\Phi\vdash\Psi$ and $\Psi\vdash\Lambda$ imply that $\Phi\vdash\Lambda$.

(as before, we use the upper-case Greek letters Φ , Ψ , Λ , ... to denote formulas, elements of $\mathcal{F}$). In other words, $\vdash$ is a *preorder* on $\mathcal{F}$. However, it is not an order: it is possible that $\Phi\vdash\Psi$ and $\Psi\vdash\Phi$, but $\Phi\neq\Psi$. For instance, when $\Phi=\forall xRxx$ and $\Psi=\forall yRyy$, this is the case. However, we feel that if $\Phi\vdash\Psi$ and $\Psi\vdash\Phi$ both hold then Φ and Ψ are "essentially the same".

The way we make this precise is that we define a relation, denoted $\equiv$, on formulas, by writing

$$\Phi\equiv\Psi \quad \underset{\text{def}}{\Longleftrightarrow} \quad \Phi\vdash\Psi \text{ and } \Psi\vdash\Phi \ .$$

We say that Φ and Ψ are *logically equivalent* if $\Phi\equiv\Psi$ holds. For instance, $\forall xRxx$ and $\forall yRyy$ are logically equivalent. There are logically equivalent formulas that look entirely different; in fact, it is in general *very difficult* to decide if two formulas are logically equivalent or not.

We can see (in fact, just by using that $\vdash$ is a preorder) that $\equiv$ is an *equivalence relation*. Next, we consider the set of all equivalence classes $[\Phi]$ of this equivalence relation. Let us denote this set by $\mathcal{F}/\equiv$. An element of $\mathcal{F}/\equiv$ is a set of the form $[\Phi]$ with Φ an arbitrary element of $\mathcal{F}$; here, $[\Phi]=\{\Psi\in\mathcal{F}:\Phi\equiv\Psi\}$. Intuitively, we "identify" two formulas when they are logically equivalent. This is very similar to modular arithmetic, in which, after a fixed modulus n is given, one *identifies* any two integers a and b , one pretends that they are equal, if $a\equiv b \pmod{n}$.

It is easy to see that the above move turns the preorder $(\mathcal{F},\vdash)$ into a (partial) order. That is, we can define $[\Phi]\leq[\Psi]$ to mean that $\Phi\vdash\Psi$, and, under this definition, $(\mathcal{F}/\equiv,\leq)$ is an order. More is true. $(\mathcal{F}/\equiv,\leq)$ *is a Boolean algebra*; in fact, in a natural way, because we have

$$[\Phi] \wedge [\Psi] = [\Phi\wedge\Psi] \;;$$
$$[\Phi] \vee [\Psi] = [\Phi\vee\Psi] \;;$$
$$-[\Phi] = [\neg\Phi]$$
the top element of $(\mathcal{F}/\equiv, \leq)$ equals $[\top]$;
the bottom element of $(\mathcal{F}/\equiv, \leq)$ equals $[\bot]$.

The Boolean algebra $(\mathcal{F}/\equiv, \leq)$ is called the *Lindenbaum-Tarski algebra of quantifier logic*. We will refer to it briefly as the *L-T-algebra*.

When we "do" Boolean algebra in the L-T-algebra, we do not write the symbols $[\ ,\ ]$ for equivalence class; on the other hand, instead of $\leq$, we write $\vdash$, and instead of $=$, we write $\equiv$. For instance, we have, as a special case of a general fact in Boolean algebras, that

$$\Phi \vdash \Psi \iff \Phi \wedge \neg\Psi \equiv \bot \;;$$

this is just the rule

$$x \leq y \iff x \wedge -y = \bot$$

that we know is true in any Boolean algebra.

However, the L-T-algebra has more structure than just the Boolean structure; this additional structure is related to the quantifiers. To explain this, we must explain *substitution*.

Given a formula Φ , and variables x and y , we can *substitute* y for x in Φ . Let us look at this in an example. In fact, we saw an example for this before. With the formula (5') in section 5.1:

$$\Phi(x) = \neg\mathbf{1}(x) \wedge \forall y\forall z(Pyzx\rightarrow(\mathbf{1}(y)\vee\mathbf{1}(z)) \,, \tag{3}$$

we also considered

$$\Phi(w) = \neg\mathbf{1}(w) \wedge \forall y\forall z(Pyzw\rightarrow(\mathbf{1}(y)\vee\mathbf{1}(z)) \,.$$

Here, $\Phi(w)$ is obtained by substituting w for x in $\Phi(x)$. In general, we write $\Phi[y/x]$ for the result of substituting y for x in Φ (Φ may have more free variables than just x; it is also allowed that x is not a free variable in Φ at all, in which case $\Phi[y/x]=\Phi$). However, we make certain exclusions in the definition of substitution.

First of all, when forming $\Phi[y/x]$, we substitute y *only for the free occurrences of the variable* x. For instance, if, for some reason, we had

$$\Phi = \neg\mathbf{1}(x) \wedge \forall y\forall z(Pyzx\rightarrow(\mathbf{1}(y)\vee\mathbf{1}(z)) \wedge \exists x(x=x),$$

(which uses a superfluous conjunct $\exists x(x=x)$, and is logically equivalent to the original Φ), then

$$\Phi[w/x] = \neg\mathbf{1}(w) \wedge \forall y\forall z(Pyzw\rightarrow(\mathbf{1}(y)\vee\mathbf{1}(z)) \wedge \exists x(x=x) ;$$

note that we *did not* substitute for the bound occurrences of x. Secondly, and more importantly,

the substitution may not alter the binding pattern of the formula; it may not introduce new bound occurrences.

This is the same as to say that

if in Φ, x *occurs as a free variable in the scope of a quantifier* $\exists y$ *or* $\forall y$ *with the same variable* y *that we want to substitute for* x, *the substitution is not allowed;* $\Phi[y/x]$ *is not defined at all.*

For instance, consider the formula $\exists yRxy$. The substitution $(\exists yRxy)[y/x]$ is not defined; in $\exists yRxy$, the free variable x is in the scope of $\exists y$, which is the formula Rxy. To take a more complicated example, with Φ as in (3), $\Phi[y/x]$ is not defined: x is a free variable in the scope of $\forall y$, which is the formula $\forall z(Pyzx\rightarrow(y=x\vee z=x)$. By the way, this is the reason why we wrote $\Phi(w)$ rather than $\Phi(y)$ in our original use of Φ.

We can explain this exclusion by pointing out that an "illegal" substitution has a meaning that does not correspond to the intention of substitution. Let's look at the case of $\Phi=\exists yRxy$. For illustration, take the interpretation $U=\mathbb{N}$, and R = ordinary $<$. Φ says "there is y such that $x<y$ ". When we do $(\exists yRxy)[z/x]$, we get $\exists yRzy$, expressing "there is y such that $z<y$ ", which is all right: just as the original formula $\exists yRxy$, $(\exists yRxy)[z/x]$ is also identically true in the given interpretation. Here, we used a *legal* substitution: z is not free in the scope of $\exists y$. However, if we did $(\exists yRxy)[y/x]$, which is *illegal* as we said above, the result wold be $\exists yRyy$, which means "there is y such that $y<y$ ", a *false* statement in the given interpretation, something that is not intended by the substitution.

We can also substitute an arbitrary term t for a variable x in a formula Φ ; the result is written $\Phi[t/x]$. This is defined (*legal*) only if by the substitution

no variable y appearing in t gets into the scope of a quantifier $\forall y$ or $\exists y$ with the same variable y .

However, we never substitute anything *for* a term if that term is not a variable. Similarly, one cannot use "quantifiers" $\forall t$, $\exists t$ unless t is a variable.

We can now express, first in an abstract and succinct way, later in some more detail, the main fact about quantifiers in the L-T-algebra.

For any formula Φ , and any variable x , we have

$$\forall x\Phi \equiv \bigwedge_t \Phi[t/x] ,$$

and

$$\exists x\Phi \equiv \bigvee_t \Phi[t/x] ;$$

here, t ranges over all terms such that the substitution $\Phi[t/x]$ is *legal*; $\bigwedge$ means meet of the set of all formulas after it, $\bigvee$ means join.

Remember that the elements of the L-T-algebra are, really, equivalence classes $[\Phi]$; in fact, what we have is

$$[\forall x\Phi] = \bigwedge_t [\Phi[t/x]] ,$$

and

$$[\exists x\Phi] = \bigvee_t [\Phi[t/x]] ;$$

but the way we wrote these relations first is nicer.

Now, we will write these relations in a more explicit way:

Rule of Universal Specification (US)**:**

$$\forall x\Phi \vdash \Phi[t/x] ;$$

Rule of Universal Generalization (UG)**:**

if $\Psi \vdash \Phi$, and x is not free in Ψ , then $\Psi \vdash \forall x\Phi$:

$$\frac{\Psi \vdash \Phi}{\Psi \vdash \forall x\Phi} \qquad \text{provided } x \text{ is not free in } \Psi .$$

Rule of Existential Generalization (EG)**:**

$$\Phi[t/x] \vdash \exists x\Phi ;$$

Rule of Existential Specification (ES)**:**

if $\Phi \vdash \Psi$, and x is not free in Ψ , then $\exists x\Phi \vdash \Psi$:

$$\frac{\Phi \vdash \Psi}{\exists x\Phi \vdash \Psi} \qquad \text{provided } x \text{ is not free in } \Psi .$$

It is understood that only legal substitutions can be taken. t denotes an arbitary term.

(The observant student will see that the second, more detailed, statement of, say, the rule for the universal quantifier is not obviously equivalent to what we had in the succinct form. In fact, universal generalization would, if translated directly from the succinct statement, look like this:

$$\frac{\Psi \vdash \Phi[t/x] \quad \text{for all } t}{\Psi \vdash \forall x \Phi}$$ (such that the substitution is legal)

It turns out that this is an equivalent formulation, if we also take into account Universal Specification).

The main result is that that the above rules for the quantifiers completely describe the L-T-algebra:

if $\Phi \vdash \Psi$ *, then this fact can be deduced using Boolean algebra, and the four rules for quantifiers,*

-- provided the formulas involved do not contain $=$ (equality); if they do, additional "axioms of equality" are needed.

This fact is a deep theorem, whose proof cannot be indicated here; it is Kurt Gödel's *completeness theorem for first order logic.*

Let us look at some examples for using the above rules, and Boolean algebra.

1. $\forall x \Phi \equiv \forall y(\Phi[y/x])$;

here, $\Phi[y/x]$ is assumed to be a legal substitution, and y is assumed to be *not free or bound in* $\forall x \Phi$.

(**Example for 1.:**

(i) $\forall xRxz \equiv \forall yRyz$;

but we are *not* asserting that $\forall xRxy \equiv \forall yRyy$, which is actually false; in this case, $\Phi = Ryz$, the second proviso, " y is not free in Φ ", is violated. Also:

(ii) $\forall x\exists u(Rxz\wedge Rux) \equiv \forall y\exists u(Ryz\wedge Ruy)$.)

Proof of 1. By US,

$\forall x\Phi \vdash \Phi[y/x]$;

therefore, since y is not free in $\forall x\Phi(x)$, by UG:

$\forall x\Phi \vdash \forall y(\Phi[y/x])$. (4)

Also, we have that

$\Phi[y/x][x/y] = \Phi$: (10)

this is because y is not bound in $\Phi[y/x]$, the substitution $\Phi[y/x][x/y]$ is legal, and all it does is to restore x where we had x before in Φ . Therefore, applying to $\Phi[y/x]$ what we already proved for Φ , with the roles of x and y exchanged, we get

$\forall y(\Phi[y/x]) \vdash \forall x(\Phi[y/x][x/y])$

which is, by (10), the same as

$\forall y(\Phi[y/x]) \vdash \forall x\Phi$ (11)

By (4) and (6), we have

$\forall x\Phi \equiv \forall y(\Phi[y/x])$

as desired.

2. $\forall x(\Lambda \vee \Gamma) \equiv \Lambda \vee \forall x\Gamma$ provided x is not free in Λ

(**Example for 2.:** $\forall x(\exists yRyz \vee Rxz) \equiv \exists yRyz \vee \forall xRxz$.)

Proof. 1 $\Gamma \vdash \Lambda \vee \Gamma$ by Boolean algebra;

2 $\forall x\Gamma \vdash \Gamma$ by US ($\Gamma[x/x] = \Gamma$);
3 $\forall x\Gamma \vdash \Lambda \vee \Gamma$ by 1 and 2 ($\vdash$ is transitive);
4 $\forall x\Gamma \vdash \forall x(\Lambda \vee \Gamma)$ by UG applied to 3 (x is not free in $\forall x\Gamma$);
5 $\Lambda \vdash \Lambda \vee \Gamma$ by Boolean algebra;
6 $\Lambda \vdash \forall x(\Lambda \vee \Gamma)$ by UG (x is not free in Λ , by assumption);
7 $\Lambda \vee \forall x\Gamma \vdash \forall x(\Lambda \vee \Gamma)$ by 4 and 6 , and Boolean algebra ($x\leq z$ and $y\leq z$ imply $x\vee y\leq z$);
8 $\forall x(\Lambda \vee \Gamma) \vdash \Lambda \vee \Gamma$ by US;
9 $\forall x(\Lambda \vee \Gamma)\wedge\neg\Lambda \vdash \Gamma$ by 8 and Boolean algebra: $x\leq y\vee z$ implies $x\wedge\text{-}z \leq y$: $x\leq y\vee z$ implies $x\wedge\text{-}z \leq (y\vee z)\wedge\text{-}z$, and $(y\vee z)\wedge\text{-}z = (y\wedge\text{-}z)\vee(z\wedge\text{-}z) = (y\wedge\text{-}z)\vee\top = y\wedge\text{-}z$; thus, $x\wedge\text{-}z \leq y\wedge\text{-}z \leq y$;
10 $\forall x(\Lambda \vee \Gamma)\wedge\neg\Lambda \vdash \forall x\Gamma$ by 9 and UG: x is not free in the premiss, since it is not free in $\forall x(\Lambda \vee \Gamma)$, and it is not free in $\neg\Lambda$ by assumption;
11 $\forall x(\Lambda \vee \Gamma) \vdash \Lambda \vee \forall x\Gamma$ by 10 and Boolean algebra: $x\wedge\text{-}z\leq y$ implies $x\leq z\vee y$;
12 $\forall x(\Lambda \vee \Gamma) \equiv \Lambda \vee \forall x\Gamma$ by 7 and 11.

3. $\exists x\Phi \equiv \neg\forall x\neg\Phi$

Proof. We prove that

$$\exists x\Phi \vee \forall x\neg\Phi \equiv \top \qquad (12)$$

and that

$$\exists x\Phi \wedge \forall x\neg\Phi \equiv \bot . \qquad (13)$$

By the definition of $\neg$ as complement in a Boolean algebra, this will mean that $\neg\forall x\neg\Phi \equiv \exists x\Phi$.

To see that $\exists x\Phi \vee \forall x\neg\Phi \equiv \top$, note that x is not free in $\exists x\Phi$; therefore, we may apply **2.** with $\Lambda=\exists x\Phi$, to get that $\exists x\Phi \vee \forall x\neg\Phi \equiv \forall x(\exists x\Phi \vee \neg\Phi)$. Now, $\Phi \vdash \exists x\Phi$ by ES. But, in any Boolean algebra, $x\leq y$ implies $y\vee\text{-}x=\top$. Applying this to Φ as x and $\exists x\Phi$ as y , we get $\exists x\Phi \vee \neg\Phi \equiv \top$ **.** But also, $\forall x\top \equiv \top$, since UG can be applied to $\top \vdash \top$ to get $\top \vdash \forall x\top$, which is $\forall x\top \equiv \top$ **.** Therefore,

$$\exists x\Phi \vee \forall x\neg\Phi \equiv \forall x(\exists x\Phi \vee \neg\Phi) \equiv \forall x\top \equiv \top .$$

This proves (12).

To show (13), note that $\forall x\neg\Phi \vdash \neg\Phi$, thus $\Phi\wedge\forall x\neg\Phi \vdash \Phi\wedge\neg\Phi \equiv \bot$, $\Phi\wedge\forall x\neg\Phi \vdash \bot$. This implies $\Phi \vdash \neg\forall x\neg\Phi$, since in a Boolean algebra, $x\wedge y=\bot$ iff $x\leq -y$. By EG, since x is not free in $\neg\forall x\neg\Phi$, we get that $\exists x\Phi \vdash \neg\forall x\neg\Phi$. By the same Boolean principle backwards, we get that $\exists x\Phi \wedge \neg\forall x\neg\Phi \vdash \bot$, what we wanted.

In the above deductions the steps taken were not suggested by plausible rules. We have found deductions of the entailments in question, but there was no explanation given *how* these deductions were found. There is a deep reason for this. As a matter of fact, there is a fundamental theoretical limitation here: *there cannot be any* completely algorithmic method of finding deductions to entailments. This is Church's Theorem on the undecidability of logical validity, a fundamental limitative result of Mathematical Logic.

The examples shown above are meant as illustrations of what we get when our search for a deduction is successful, but not of a method of finding a deduction.

On the other hand, there are various ways of organizing the search for a deduction that help in many practically important cases to find one. These ways belong to the subject area of Computer Science called Automatic Theorem Proving.