Show $(H \wedge K) \supset L$, where

$H = (\forall x)(\forall y)[Rxy \supset Ryx]$ *(R is symmetric)*

$K = (\forall x)(\forall y)(\forall z)[(Rxy \wedge Ryz) \supset Rxz]$ *(R is transitive)*

$L = (\forall x)(\forall y)[Rxy \supset Rxx]$ (R is *reflexive* on its domain of definition).

For a *hard* one, try the following exercise (taken from Quine [1]): Show $(A \wedge B) \supset C$, where

$A = (\forall x)[(Fx \wedge Gx) \supset Hx] \supset (\exists x)[Fx \wedge {\sim}Gx]$

$B = (\forall x)\ [Fx \supset Gx] \vee (\forall x)[Fx \supset Hx]$

$C = (\forall x)[(Fx \wedge Hx) \supset Gx] \supset (\exists x)[Fx \wedge Gx \wedge {\sim}Hx]$

§ 3. The Completeness Theorem

Now we turn to the proof of one of the major results in quantification theory: Every valid sentence is provable by the tableau method.

This is a form of Gödel's famous completeness theorem. Actually Gödel proved the completeness of a different formalization of Quantification Theorey, but we shall later show how the completeness of the tableau method implies the completeness of the more conventional formalizations. The completeness proof we now give is along the lines of Beth [1] or Hintikka [1]—and also Anderson and Belnap [1].

Let us first briefly review our completeness proof for tableaux in *propositional* logic, and then see what modifications will suggest themselves. In the case of propositional logic, we reach a completed tableau after finitely many stages. Upon completion, every open branch is a Hintikka set. And by Hintikka's lemma, every Hintikka set is truth-functionally satisfiable.

Our first task is to give an appropriate definition of "Hintikka set" for first order logic in which we specify conditions not only on the α's and β's but also on the γ's and δ's as well. We shall define Hintikka sets for arbitrary universes U of constants.

Definition. By a Hintikka set (for a universe U) we mean a set S (of U-formulas) such that the following conditions hold for every α, β, γ, δ in E^U:

H_0: No *atomic* element of E^U and its negation (or conjugate, if we are working with signed formulas) are both in S.

H_1: If $\alpha \in S$, then α_1, α_2 are both in S.

H_2: If $\beta \in S$, then $\beta_1 \in S$ or $\beta_2 \in S$.

H_3: If $\gamma \in S$, then for *every* $k \in U, \gamma(k) \in S$.

H_4: If $\delta \in S$, then for at least one element $k \in U$, $\delta(k) \in S$.

Now we show

Lemma *(Hintikka's lemma for first order logic). Every Hintikka set S for a domain U is (first order) satisfiable—indeed in the domain U.*

Proof. We must find an *atomic* valuation of E^U in which all elements of S are true. We do this exactly as we did propositional logic—viz for every *atomic* sentence $P\xi_1,\ldots,\xi_n$ of E^U, give it the value t if $TP\xi_1,\ldots,\xi_n$ is an element of S, f if $FP\xi_1,\ldots,\xi_n$ is an element of S, and either t or f at will if neither is an element of S. We must show that each element X of S is true under this atomic valuation. Again we do this by induction on the degree of X. If X is of degree 0, it is immediate that X is true (under this valuation). Now suppose that X is of positive degree and that every element of S of lower degree is true. We must show X is true. Since X is not of degree 0, then it is either some α, β, γ or δ. If it is an α or a β, then it is true for exactly the same reasons as in the proof of Hintikka's lemma for propositional logic (viz. if it is an α, then α_1, α_2 are both in S, hence both true (by induction hypothesis), hence α is true; if it is a β, then at least one of β_1, β_2 is in S, and hence true, so β is true). Thus the new cases to consider are γ, δ.

Suppose it is a γ. Then, for *every* $k\in U$, $\gamma(k)\in S$ (by H_3), but every $\gamma(k)$ is of lower degree than γ, hence true by inductive hypothesis. Hence γ must be true.

Suppose it is a δ. Then for at least one $k\in U$, $\delta(k)\in S$ (by H_4). Then $\delta(k)$ is true by inductive hypothesis, hence δ is true.

We next consider how we can use Hintikka's lemma for our completeness proof. In propositional logic, tableaux terminate after finitely many steps. But a tableau for first order logic may run on infinitely without ever closing. Suppose this should happen. Then we generate an infinite tree $\mathscr{T}$, and by König's Lemma, $\mathscr{T}$ contains an infinite branch θ. Clearly θ is open, but do the elements of θ necessarily constitute a Hintikka set? The answer is "no" as the following considerations will show.

For any X on a branch θ of degree >0, define X to be *fulfilled* on θ if either: (*i*) X is an α, and α_1, α_2 are both on θ; (*ii*) X is a β and at least one of β_1, β_2 is on θ; (*iii*) X is a γ and for *every* parameter a, $\gamma(a)$ is on θ; (*iv*) X is a δ and for at least one parameter a, $\delta(a)$ is on θ. Now suppose $\mathscr{T}$ is a finite tableau and that a branch contains two γ-sentences—call them γ_1 and γ_2. Now suppose we use γ_1 and successively adjoin $\gamma_1(a_1)$, $\gamma_1(a_2),\ldots,\gamma_1(a_n),\ldots$, for *all* the parameters $a_1,a_2,\ldots,a_n,\ldots$. We thus generate an infinite branch and we have clearly taken care of fulfilling γ_1, but we have totally neglected γ_2. Or it is possible to fulfill a γ-formula on a branch but neglect one or several α, β, or δ formulas on the branch. Thus there are many ways in which an infinite tableau can be generated without all—or even any—open branches being Hintikka sets. The key problem is to find a *systematic* procedure which will guarantee that any tableau constructed according to the procedure is such that if it runs on infinitely, every open branch will have to be a Hintikka sequence.

Many such procedures exist in the literature; the reader should at this point try to work out such a procedure for himself before reading further.

The following systematic procedure seems to be as simple and direct as any. In this procedure of generating the tree, at each stage certain points of the tree are declared to have been "used" (as a practical book-keeping device, we can put a check mark to the right of a point of the tableau as soon as we have used it).

Now for a precise description of the procedure. We start the tableau by placing the formula (whose satisfiability we are testing) at the origin. This concludes the first stage. Now suppose we have concluded the nth stage. Then our next act is determined as follows. If the tableau already at hand is closed, then we stop. Also, if every non-atomic point on every open branch of the tableau at hand has been used, then we stop. If neither, then we pick a point X of *minimal* level (i.e. as high up on the tree as possible) which has not yet been used and which appears on at least one open branch[1]), and we extend the tableau at hand as follows: we take *every* open branch θ passing through the point X, and

1) If X is an α, we extend θ to the branch $(\theta, \alpha_1, \alpha_2)$.

2) If X is a β then we simultaneously extend θ to the 2 branches (θ, β_1) and (θ, β_2).

3) If X is a δ then we take the first parameter a which does not appear on the tree and we extend θ to $(\theta, \delta(a))$.

4) If X is a γ (and this is the delicate case!), then we take the first parameter a such that $\gamma(a)$ does not occur on θ, and we extend θ to $(\theta, \gamma(a), \gamma)$. (In other words we add $\gamma(a)$ as an endpoint to θ and then we *repeat* an occurrence of γ!)

Having performed acts 1—4 (depending on whether X is respectively an α, β, γ, δ), we then declare the point X to be used, and this concludes the stage $n+1$ of our procedure.

Discussion. To describe the above procedure more informally, we *systematically* work our way down the tree, automatically fulfilling all α, β and δ formulas which come our way. As to the γ-formulas, when we use an occurrence of γ on a branch θ to subjoin an instance $\gamma(a)$, the purpose of *repeating* an occurrence of γ is that we must sooner or later come down the branch θ and use this repeated occurrence, from which we adjoin another instance $\gamma(b)$ and repeat an occurrence of γ

---

[1]) If the reader wishes to make the procedure completely deterministic he can, e.g. pick the *leftmost* such unused point of minimal level.