1 First-Order Uniform Notation

In Lecture 25 we introduced the rules of Semantic Tableaux by classifying all quantified first-order formulae and their negations into two groups, those that act *universally*, which are called *type-C formulae*, and those that act *existentially*, which are called *type-D formulae*. The groups and notation is given in the following table (where t is any variable-free term).

type-C (universal)	
C	$C(t)$
$(\forall x)\phi(x)$	$\phi(t)$
$\neg\,(\exists x)\phi(x)$	$\neg\,\phi(t)$

type-D (existential)	
D	$D(t)$
$(\exists x)\phi(x)$	$\phi(t)$
$\neg\,(\forall x)\phi(x)$	$\neg\,\phi(t)$

If γ is a type-C formula, then we will use the convention that $\gamma(x)$ is the component formula (where x does not occur free in γ). For example, if $\gamma = (\forall x)\phi$, then $\gamma(x) = \phi(x)$. We then write $\gamma(t)$ for the result of substituting the ground term t for x in the component γ. If $\gamma = (\forall x)\phi$, then $\gamma(t) = \phi^x_t$. Similarly, if δ is a type-D formula, then $\delta(x)$ is the component formula (where x does not occur free in δ). For example, if $\delta = (\exists x)\phi$, then $\delta(x) = \phi(x)$. We then write $\delta(t)$ for the result of substituting the ground term t for x in the component δ. If $\delta = (\exists x)\phi$, then $\delta(t) = \phi^x_t$.

Essentially, all type-C formulae act universally and all type-D formulae act existentially. More precisely, if γ is a type-C formulae, then $\gamma \leftrightarrow (\forall x)\gamma(x)$ is valid, and if if δ is a type-D formulae, then $\delta \leftrightarrow (\exists x)\delta(x)$ is valid. In terms of the four cases above, this means that the following first-order formulae are valid (that is, their universal closures are valid):

$$(\forall x)\phi \leftrightarrow (\forall x)\phi \qquad \neg\,(\exists x)\phi \leftrightarrow (\forall x)\neg\,\phi$$
$$(\exists x)\phi \leftrightarrow (\exists x)\phi \qquad \neg\,(\forall x)\phi \leftrightarrow (\exists x)\neg\,\phi$$

These equivalences can be verified by the truth conditions for quantified formulae:

$$v_{\mathcal{A}}((\exists x)\phi) = \begin{cases} \mathbf{T} & \text{if } v_{\mathcal{A}}(\phi^x_a) = \mathbf{T} \text{ for at least one } a \in A, \\ \mathbf{F} & \text{if } v_{\mathcal{A}}(\phi^x_a) = \mathbf{F} \text{ for all } a \in \mathcal{A}; \end{cases}$$
$$v_{\mathcal{A}}((\forall x)\phi) = \begin{cases} \mathbf{T} & \text{if } v_{\mathcal{A}}(\phi^x_a) = \mathbf{T} \text{ for all } a \in \mathcal{A}, \\ \mathbf{F} & \text{if } v_{\mathcal{A}}(\phi^x_a) = \mathbf{F} \text{ for at least one } a \in A; \end{cases}$$

The following theorem explains why our rules for type-C and type-D formulae in semantic tableaux are sound. Recall that a *ground term* is a constant of the language. (The following theorem would still hold if we had function symbols in the language, so more complex ways of constructing terms.)

Theorem 29.1.1 Let Σ be a set of sentences, γ a type-C sentence and δ a type-D sentence.

1. If $\Sigma \cup \{\gamma\}$ is satisfiable, then so is $\Sigma \cup \{\gamma, \gamma(t)\}$ for any ground term.
2. If $\Sigma \cup \{\delta\}$ is satisfiable, then so is $\Sigma \cup \{\delta, \delta(c)\}$ provided c is a constant new to Σ and δ.

Proof.
(1). Let $\Sigma \cup \{\gamma\}$ be satisfied in a structure $\mathcal{A}$, and let $t^{\mathcal{A}} = a$. Since $\gamma = (\forall x)\gamma(x)$ is valid, $\gamma(a)$ is true in $\mathcal{A}$. Equivalently, $\gamma(t)$ is true in $\mathcal{A}$.

(2). Let $\Sigma \cup \{\delta\}$ be satisfied in a structure $\mathcal{A}$. Since $\delta \leftrightarrow (\exists x)\delta(x)$ is valid, it is true in $\mathcal{A}$. So, there is some element $a \in A$ for which $\delta(a)$ is true. Let c be a constant symbol new to the language $\mathcal{L}$ and extend the structure $\mathcal{A}$ so that $c^{\mathcal{A}} = a$. Then $\delta(c)$ is true in this extension of $\mathcal{A}$, and each of the sentences of Σ and δ remain true in the extension by Lemma 22.3.3. □

2 Structural Induction and Recursion using Unified Notation

We can extend the notion of structural induction and recursion to make use of first-order unified notation, just as we did for propositional logic in Lecture 7. The form given here is more closely related the structural induction and recursion in the extended sense of Theorem 21.3.4 and 21.3.5.

Theorem 29.2.1 (First-Order Structural Induction) Let X be a set of formulae of a first-order language $\mathcal{L}$ (possibly extended by new constants). Then X is the set of all first-order formulae of $\mathcal{L}$, provided:

Basis Step. Each atomic formulae and its negation is in X.

Induction Steps.

(a) If α is a type-A formulae with whose components α_1 and α_2 are in X, then α is in X as well.

(b) If β is a type-B formulae with whose components β_1 and β_2 are in X, then β is in X as well.

(c) If γ is a type-C formulae and each of the formulae $\gamma(t)$ is in X, for each term t of the language, then γ is in X as well.

(d) If δ is a type-D formulae and each of the formulae $\delta(t)$ is in X, for each term t of the language, then δ is in X as well.

Theorem 29.2.2 (First-Order Structural Recursion) There is one and only one function F defined on the set of formulae for a language $\mathcal{L}$ such that

Basis Step. The value of F is specified explicitly on atomic formulae and their negation.

Induction Steps.

(a) The value of F on type-A formulae α is specified in terms of the values of F on its components α_1 and α_2.

(b) The value of F on type-B formulae β is specified in terms of the values of F on its components β_1 and β_2.

(c) The value of F on type-C formulae γ is specified in terms of the values of F on its components $\gamma(t)$ for each term t.

(d) The value of F on type-D formulae δ is specified in terms of the values of F on its components $\delta(t)$ for each term t.

3 Hintikka Sets

Remark 29.3.1 Just as in the case of propositional logic, semantic tableaux provide a means of constructing Hintikka sets, which guarantee the existence of a satisfying structure. We will show here that every Hintikka set is satisfiable, and then later show that every finished noncontradictory branch in a semantic tableaux is a Hintikka set, so satisfiable.

We built semantic tableaux by extending a language $\mathcal{L}$ to a language with an infinite set $A = \{a_1, a_2, \ldots\}$ of parameters (new constants). When we built a structure from a finished noncontradictory branch we used the parameters to form a domain along with any constants occurring in $\mathcal{L}$. It will be convenient to assume that the constants of $\mathcal{L}$ are included in the parameters of A (or at least that we have fixed an interpretation $c^{\mathcal{A}} \in A$). For example, we could reserve the even numbered parameters, $\{a_2, a_4, \ldots\}$, for interpreting constant symbols of $\mathcal{L}$ and odd numbered parameters, $\{a_2, a_4, \ldots\}$, as uninterpreted parameters

that we use in building tableaux. We will define Hintikka sets for a language $\mathcal{L}$ over arbitrary universes A of constants and assume that the constants of $\mathcal{L}$ are either included (or interpreted) in this domain.

Definition 29.3.2 (Hintikka set) Let $\mathcal{L}$ be a first-order language and A a nonempty set of parameters. A set S of sentences in $\mathcal{L}^A$ is called a *Hintikka set* (over the domain A) if it satisfies the following conditions

(H0) No *atomic* sentence and its negation are both in S.

(H1) If α is in S, then both of its components α_1 and α_2 are in S.

(H2) If β is in S, then at least one of its components β_1 or β_2 is in S.

(H3) If γ is in S, then $\gamma(a)$ is also in S for every $a \in A$.

(H4) If δ is in S, then then $\delta(a)$ is also in S for at least one $a \in A$.

Remark 29.3.3 We now prove that every Hintikka set S for a domain A is satisfiable. The challenge here is where we are to find such a structure, when the only available material are the sentences in S. The idea is to use the parameters of A themselves to furnish the domain. This idea has wide repercussions and the basis of several very important results, such the Löwenheim-Skolem Theorem and the Model Existence Theorem to follow.

Lemma 29.3.4 (Hintikka's Lemma) Let $\mathcal{L}$ be a first-order language. Every Hintikka set for $\mathcal{L}$ over a nonempty domain A is satisfiable in a structure whose domain is A.

Proof. Let S be a Hintikka set over the domain A. We will define a structure $\mathcal{A}$ with domain A as follows.

For each $a \in A$, let $a^{\mathcal{A}} = a$. (Note that by the above remark the constant symbols in $\mathcal{L}$ are either in A or are already interpreted in A.)

For each n-ary relation symbol R, let

$$R^{\mathcal{A}} = \big\{\langle a_1, \ldots, a_n\rangle \,:\, R(a_1, \ldots, a_n) \in S\big\}.$$

We will show by structural induction that for sentence $\phi \in S$

$$\alpha \in S \quad \text{implies} \quad v_{\mathcal{A}}(\phi) = \mathbf{T}.$$

This is true for atomic sentences by (H0) and the definition of the interpretations $R^{\mathcal{A}}$. For example, if R is an n-ary and $a_1, \ldots, a_n$ are any terms from A,

$$\begin{aligned} v_{\mathcal{A}}\big(R(a_1, \ldots, a_n)\big) = \mathbf{T} \quad &\text{iff} \quad \langle a_1^{\mathcal{A}}, \ldots, a_n^{\mathcal{A}}\rangle \in R^{\mathcal{A}} \\ &\text{iff} \quad \langle a_1, \ldots, a_n\rangle \in R^{\mathcal{A}} \\ &\text{iff} \quad R(a_1, \ldots, a_n) \in S. \end{aligned}$$

The case where ϕ is type-A or type-B is exactly the same as the propositional case Lemma 13.1.4. Suppose $\phi = \gamma$ is type-C and that $\gamma(a) \in S$ implies $v_A(\gamma(a)) = \mathbf{T}$ for each $a \in A$ (inductive hypothesis).

$$\begin{aligned} \gamma \in S \quad &\text{implies} \quad \gamma(a) \in S \qquad \text{for all } a \in A \\ &\text{implies} \quad v_{\mathcal{A}}(\gamma(a)) = \mathbf{T} \qquad \text{for all } a \in A \\ &\quad\;\; \text{iff} \quad v_{\mathcal{A}}((\forall x)\gamma(x)) = \mathbf{T} \\ &\quad\;\; \text{iff} \quad v_{\mathcal{A}}(\gamma) = \mathbf{T} \end{aligned}$$

The first line is by condition (H3), the second line by the inductive hypothesis, and the last line is because $\gamma \leftrightarrow (\forall x)\gamma(x)$ is valid,

Suppose $\phi = \delta$ is type-D and that $\gamma(a) \in S$ implies $v_A(\gamma(a)) = \mathbf{T}$ for each $a \in A$ (inductive hypothesis). Since $\delta \leftrightarrow (\exists x)\delta(x)$ is valid,

$$\begin{array}{lll} \delta \in S & \text{implies} & \delta(a) \in S \qquad \text{for at least one } a \in A \\ & \text{implies} & v_{\mathcal{A}}(\delta(a)) = \mathbf{T} \qquad \text{for at least one } a \in A \\ & \text{iff} & v_{\mathcal{A}}((\exists x)\delta(x)) = \mathbf{T} \\ & \text{iff} & v_{\mathcal{A}}(\delta) = \mathbf{T} \end{array}$$

The first line is by condition (H4), the second line by the inductive hypothesis, and the last line is because $\delta \leftrightarrow (\exists x)\delta(x)$ is valid,

□