PHIL 151 First-Order Logic

Recursion

Shane Steinert-Threlkeld

January 20, 2012

Definition 1. *An* operation *on a set U is a function $f : U^n \to U$ for some $n \geq 1$.*

We denote n by $\#f$ and call this the arity *of f.*

Recall from Wednesday's lecture that we can define the smallest inductive set containing certain basis elements and closed under the operations in a family. Let $\mathcal{F}$ refer to a family of operations on U. We can redefine a *construction sequence* to be a sequence $\langle x_1, \ldots, x_n \rangle$ such that for each $i \leq n$, at least one of the following holds:

$$x_i \in B$$

$$x_i = f(x_{j_1}, \ldots, x_{j_m}) \qquad f \in \mathcal{F},\ \#f = m,\ j_1, \ldots, j_m < i$$

Definition 2. *$\bigcup_n C_n = \bigcap \{S \mid S \text{ is inductive}\}$ where C_n is the set of conclusions of construction sequences of length $\leq n$ and a set S is inductive if $B \subseteq S$ and S is closed under every $f \in \mathcal{F}$.*

We call this C, the set generated from B *by the operations in $\mathcal{F}$.*

Inductive sets allow us to do some neat things. For one, we can do proof by induction considering cases corresponding to every operation in $\mathcal{F}$. Mathematical induction on the natural numbers and structural induction on the well-formed formulas are just two examples of this. But we'd also like to be able to define functions recursively.

The general template for a recursive definition of a function h runs as follows:

- Define $h(x, \vec{y})$ for $x \in B$.
- For every $f \in \mathcal{F}$, define $h(f(x_1, \ldots, x_{\#f}), \vec{y})$ in terms of $h(x_1, \vec{y}), \ldots, h(x_{\#f}, \vec{y})$.

(Note that $\vec{y}$ is a list of additional parameters from U which may be empty.) While we do require that $\mathrm{dom}(h) = U^n$ for some $n \geq 1$, h can in fact be a function going to any arbitrary set.

Recalling that we can consider $\mathbb{N}$ as the set generated from $B = \{0\}$ by $\mathcal{F} = \{S\}$ where $S(x) = x + 1$, consider the following definitions of functions:

- $h(0, y) = y$
- $h(Sx, y) = S(h(x, y))$

Can anyone guess what function h defines?

- $h(0, y) = 0$
- $h(Sx, y) = y + h(x, y)$

Can anyone guess what function h defines?

- $h(0) = 1$
- $h(Sx) = Sx \cdot h(x)$

Can anyone guess what function f defines?

Exercise 1. *Give a recursive definition of exponentiation* $h(n, m) = n^m$*.*

Give a recursive definition of a function $h(n)$ *that computes the* n^{th} *Fibonacci number. (*$1, 1, 2, 3, 5, 8, 13, 21, \ldots$*)*

Remark 1. *Were we to introduce one more operation,* bounded minimization*, and define a small set of "initial operations" that contains even less than addition and multiplication, we would generate what are known as the* primitive recursive functions*. These functions play a critical role in computability theory; you will learn a lot about them if you go on to take 152. Also note that the primitive recursive functions define a programming language containing bounded for loops (but no while loops).*

Now, there can be more than one way of generating an element in C. For instance, if we consider $\mathbb{Z}$ as generated by successor, S, and predecessor, P, we see that $2 = SS0 = SSPS0$. When we have distinct construction sequences for elements in our inductive set, there is no guarantee that a function defined by recursion will exist. For an example, see Enderton page 38.

In order to admit recursive definitions, we need a stronger notion than generation:

Definition 3. *A set C is* freely *generated from B by $\mathcal{F}$ if it is generated from B by $\mathcal{F}$ and, furthermore,*

1. *Each $f \in \mathcal{F}$ is one-to-one.*
2. *For every $f, g \in \mathcal{F}$, $ran(f) \cap ran(g) = \emptyset$.*
3. *For every $f \in \mathcal{F}$, $ran(f) \cap B = \emptyset$.*

Lemma 1. *The natural numbers, $\mathbb{N}$, are freely generated from $\{0\}$ by $\{S\}$.*

Proof. Exercise. □

Theorem 1 (Unique Readability)**.** *WFF is freely generated from $A = \{A_1, A_2, \ldots\}$ by $\mathcal{E} = \{\mathcal{E}_\neg, \mathcal{E}_\vee, \mathcal{E}_\wedge, \mathcal{E}_\to, \mathcal{E}_\leftrightarrow\}$.*

Proof. By the definition of free generation, we need to show that every operation in $\mathcal{E}$ is one-to-one, disjoint from A, and disjoint from every other operation.

Consider the case of $\mathcal{E}_\wedge$. To show that it is one-to-one, suppose that

$$(\alpha \wedge \beta) = (\gamma \wedge \delta)$$

We need to show that $\alpha = \gamma$ and $\beta = \delta$. We delete the first (to get

$$\alpha \wedge \beta) = \gamma \wedge \delta)$$

By Lemma 13B, in §1.3, which can be read on your own, $\alpha = \gamma$. Deleting the $\wedge$ and applying the same lemma yields that $\beta = \delta$.

Similar considerations show that the other operations in $\mathcal{E}$ are one-to-one.

To show that ran($\mathcal{E}_\wedge$) is disjoint from the ranges of the other operations, consider $\mathcal{E}_\to$. Suppose, for the purpose of deriving a contradiction, that the ranges are not disjoint, i.e. that for some wffs

$$(\alpha \wedge \beta) = (\gamma \to \delta)$$

By the same reasoning as earlier, $\alpha = \gamma$. Moreover, we have that $\wedge = \to$, a contradiction. The other cases are similar.

To show that $\mathrm{ran}(\mathcal{E}_\Box) \cap A = \emptyset$ for every $\mathcal{E}_\Box \in \mathcal{E}$, simply note that every formula in each range begins with (, while no element of A does. $\Box$

Non-example. So our two canonical examples of generated / inductive sets are also freely generated. A quick non-example is that $\mathbb{Z}$ is not freely generated from $\{0\}$ by $\{S, P\}$. In particular, $\mathrm{ran}(S) \cap \mathrm{ran}(P) \neq \emptyset$. These ranges are also not disjoint from $\{0\}$.

Theorem 2 (Recursion Theorem)**.** *Suppose that $C \subseteq U$ is freely generated from B by a family of operations $\mathcal{F}$.*

Assume that V is a set, and that we have functions

$$\begin{aligned} h &: B \times U^n \to V \\ F &: V^{\#f} \to V \end{aligned}$$

for every $f \in \mathcal{F}$ and some $n \geq 0$.

Then there is a unique function

$$\bar{h} : C \times U^n \to V$$

such that

1. $\bar{h}(x, \vec{y}) = h(x, \vec{y})$ *for every* $x \in B$
2. $\bar{h}(f(x_1, \dots, x_{\#f}), \vec{y}) = F(\bar{h}(x_1, \vec{y}), \dots, \bar{h}(x_{\#f}, \vec{y}))$ *for every* $f \in \mathcal{F}$.

Let's recast some of our earlier examples in light of this new definition. Consider addition:

- $h(0, y) = y$
- $h(Sx, y) = S(h(x, y))$

In this case, we have (in addition to $B = \{0\}$, $\mathcal{F} = \{S\}$):

- $V = \mathbb{N}$
- $n = 1$
- $s : V^{\#S} \to V = S$, i.e. what we're already given.

For the multiplication and factorial cases, we also used functions previously defined by recursion (addition and multiplication, respectively). We could replace these with the original definitions to arrive at a completely strict definition by recursion, but it would lose readability and comprehensibility. Do note that in the factorial case, $n = 0$.

In the case most pertinent to this course, we can now apply the two main theorems of today to prove that a truth assignment on sentence symbols can be uniquely extended to one on all wff's satisfying the conditions at the beginning of §1.2.

Theorem 3. *A truth assignment $v : A \to \{T, F\}$ can be extended uniquely to an assignment $\bar{v} : WFF \to \{T, F\}$ such that conditions (1)-(5) are satisfied.*

Proof. First, we define 5 new functions. Let $N : \{T, F\} \to \{T, F\}$ be given by

$$N(x) = \begin{cases} T & x = F \\ F & x = T \end{cases}$$

Let $K, A, C, E : \{T, F\} \times \{T, F\} \to \{T, F\}$ be given by:

$$K(x, y) = \begin{cases} T & x = y = T \\ F & \text{otherwise} \end{cases}$$

$$A(x, y) = \begin{cases} T & x = T \text{ or } y = T \\ F & \text{otherwise} \end{cases}$$

$$C(x, y) = \begin{cases} T & \text{otherwise} \\ F & x = T \text{ and } y = F \end{cases}$$

$$E(x, y) = \begin{cases} T & x = y \\ F & \text{otherwise} \end{cases}$$

By the unique readability theorem, WFF is freely generated from A by $\mathcal{E}$. Therefore, by the recursion theorem, given $v : A \to \{T, F\}$, there is a unique $\bar{v} : WFF \to \{T, F\}$ such that

$$\begin{aligned} \bar{v}(\neg\alpha) &= \bar{v}(N(\alpha)) \\ \bar{v}(\alpha \wedge \beta) &= K(\bar{v}(\alpha), \bar{v}(\beta)) \\ \bar{v}(\alpha \vee \beta) &= A(\bar{v}(\alpha), \bar{v}(\beta)) \\ \bar{v}(\alpha \to \beta) &= C(\bar{v}(\alpha), \bar{v}(\beta)) \\ \bar{v}(\alpha \leftrightarrow \beta) &= E(\bar{v}(\alpha), \bar{v}(\beta)) \end{aligned}$$

Substituting the right-hand side for the definitions of N, A, O, C, E gives the conditions (1)-(5) as desired. $\square$

I will now sketch a proof of the Recursion Theorem, defining a few notions more explicitly than Enderton. Nevertheless, the details are tedious and would take too much class time. Feel free to read the full proof on pages 41 - 44.

Proof.

Definition 4. *A function v is* acceptable *if $dom(v) \subseteq C$, $ran(v) \subseteq V$, and for any $x, y \in C$:*

1. $x \in B \cap dom(v) \Rightarrow v(x) = h(x)$
2. *For any $f \in \mathcal{F}$, if $f(x_1, \ldots, x_{\#f}) \in dom(v)$, then $v(f(x_1, \ldots, x_{\#f})) = F(v(x_1), \ldots, v(x_{\#f}))$.*

The idea is that an acceptable function approximates a recursive function. It satisfies the same conditions that we want our recursive extension of h to satisfy, but may not be defined on the whole of C.

Definition 5. *The graph of a function $f : C \to V$, denoted $gr(f)$, is the set of ordered pairs*

$$gr(f) = \{\langle x, f(x)\rangle \mid x \in dom(f)\}$$

Note that Enderton just identifies a function with its graph; for reasons that I won't go in to here, I think it's useful to keep the two notions conceptually distinct.

Now, let $K = \{\mathrm{gr}(v) \mid v \text{ is acceptable}\}$. Define $\bar{h} = \bigcup K$. That is, $\langle x, y\rangle \in \bar{h}$ iff there is an acceptable v such that $\langle x, y\rangle \in \mathrm{gr}(v)$, i.e. iff $v(x) = y$ for some acceptable function v.

This $\bar{h}$ will in fact be (the graph of) our unique extension. Showing this requires four steps:

1. $\bar{h}$ is in fact (the graph of) a function. For every $x \in \mathrm{dom}(\bar{h})$, there is a *unique* y such that $\langle x, y\rangle \in \bar{h}$. Do this by showing that

   $$S = \left\{x \in C \mid \text{for at most one } z, \langle x, z\rangle \in \bar{h}\right\}$$

   is an inductive set.

2. $\bar{h} \in K$

3. $\mathrm{dom}(h) = C$. Do this by showing that $\mathrm{dom}(h)$ is an inductive set.

4. $\bar{h}$ is unique. Given another function satisfying (1)-(3), the set on which the two agree will be inductive, i.e. all of C.

$\square$

Recall that in our proof of the uniqueness of the extended truth assignment $\bar{v}$, we introduced functions $N : \{T, F\} \to \{T, F\}$ and $K, A, C, E : \{T, F\}^2 \to \{T, F\}$. These functions motivate a definition used throughout §1.5 and a bit beyond:

Definition 6. *A* Boolean function *is an operation on* $\{T, F\}$. *In other words, a* Boolean function *is a function* $f : \{T, F\}^n \to \{T, F\}$ *for some* n.

To every wff we can define a corresponding Boolean function:

Definition 7. *Suppose* $\alpha \in WFF$ *has sentence symbols at most* $A_1, \ldots, A_n$. *Then the* Boolean function realized by α, B^n_α, *is given by*

$$B^n_\alpha(x_1, \ldots, x_n) = \bar{v}(\alpha)$$

where $\bar{v}$ is the unique extension of the truth assignment $v(A_i) = x_i$.

Seen in this light, what we showed in the proof of the uniqueness of the extension of a truth assignment is that:

$$N = B^1_{\neg A_1}$$

$$K = B^2_{A_1 \wedge A_2}$$

$$A = B^2_{A_1 \vee A_2}$$

$$C = B^2_{A_1 \rightarrow A_2}$$

$$E = B^2_{A_1 \leftrightarrow A_2}$$

We will now see that any Boolean function is realized by a WFF in our sentential language. Moreover, we can actually prove the form of such formulas. This notion, which I will now define, of disjunctive normal form plays a crucial role in automated reasoning.

Definition 8. *A* literal *is a sentence symbol or the negation of a sentence symbol.*

Definition 9. *A wff α is in* disjunctive normal form (dnf) *if*

$$\alpha = \gamma_1 \vee \cdots \vee \gamma_k$$

where each γ_i is a conjunction of literals β_{ij}:

$$\gamma_i = \beta_{i1} \wedge \cdots \wedge \beta_{in_i}$$

Theorem 4. *Let G be an n-place Boolean function. There is a wff α in disjunctive normal form such that $G = \beta^n_\alpha$.*

Proof. We argue by cases.

Case 1: $G = F$ for all $\vec{x} \in \{T, F\}^n$. Then $G = B^1_{A_1 \wedge \neg A_1}$ where $\alpha = A_1 \wedge \neg A_1$ is in dnf.

Case 2: $G(\vec{x}) = T$ for k vectors in $\{T, F\}^n$. Enumerate them:

$$\begin{aligned}
\vec{x_1} &= \langle x_{11}, x_{12}, \ldots, x_{1n} \rangle \\
\vec{x_2} &= \langle x_{21}, x_{22}, \ldots, x_{2n} \rangle \\
&\ \vdots \\
\vec{x_k} &= \langle x_{k1}, x_{k2}, \ldots, x_{kn} \rangle
\end{aligned}$$

Now, define

$$\begin{aligned}
\beta_{ij} &= \begin{cases} A_j & x_{ij} = T \\ (\neg A_j) & x_{ij} = F \end{cases} \\
\gamma_i &= \beta_{i1} \wedge \cdots \wedge \beta_{in} \\
\alpha &= \gamma_1 \vee \gamma_2 \vee \cdots \vee \gamma_k
\end{aligned}$$

First, note that by construction, α is in dnf. We now show that $G = B^n_\alpha$.

Note that for all $1 \leq i \leq k$, $B^n_\alpha(\vec{x_i}) = T$. This is because $B^n_{\gamma_i}(\vec{x_i}) = T$ for each i. Now, because γ_i is a conjunction of n literals of the sentence symbols $\{A_1, \ldots, A_n\}$, only one truth assignment on $\{A_1, \ldots, A_n\}$ can satisfy γ_i. Since α is a disjunction of k γ_is, there are exactly k truth-value assignments that can satisfy α. As we have seen, these are exactly the ones corresponding to the k vectors $\vec{x_i}$. Therefore, $B^n_\alpha(\vec{x}) = G(\vec{x})$ for all $\vec{x} \in \{T, F\}^n$. □

Exercise 2. *Define* conjuctive normal form (cnf) *analogously to disjunctive normal form: α is in cnf iff*

$$\alpha = \gamma_1 \wedge \cdots \wedge \gamma_k$$

where $\gamma_i = \beta_{i1} \vee \cdots \vee \beta_{in}$ is a disjunction of literals.

Prove that for any n-place Boolean function G, we can find a wff α in cnf such that $G = B^n_\alpha$.

Note that the theorem and the exercise can be re-read as: for any wff φ, there is a tautologically equivalent wff α in dnf (resp. cnf).