As I said, Frege’s universal and existential quantifiers are somewhat unnatural. But even worse is the fact that many quantifiers can’t be represented at all with good old $\forall$ and $\exists$:

- Most $A$ are $B$: “Most people who see ghosts are somewhat suggestible.”

- “At least 5 $A$ are $B$”: “Somebody at the party infected at least five people.”

\((\exists x (at(x, party201) \land \exists y (during(party201, infect(x, y))))), to be a quite hand-wavy about it.\)

Now for what seems like a digression but isn’t. It’s an interesting observation that we really need only one variable-binding operator: $\lambda$. We can write, e.g., $\forall x P$ as $\forall (\lambda x. P)$. But then what is $\forall$? It looks like a predicate on predicates. And sure enough, it’s easy to state the truth conditions for $\forall$ and $\exists$ in their new incarnation:

- $(D, I) \models g \forall P$ iff $I_g(P) = D$

- $(D, I) \models g \exists P$ iff $I_g(P) \neq \emptyset$

Recall that $I_g$ is the interpretation of a formula with respect to variable assignment $g$. $I_g(P)$, where $P$ is a predicate, is a set of objects in the domain, i.e., a subset of $D$. So $(D, I) \models g \exists(\text{red})$ if $I_g(\text{red})$ is not the the empty set, i.e., if there are objects in the set of everything red. Or we could be even more direct. If $\forall$ and $\exists$ are really predicates, they should have interpretations that are sets of . . . what? Sets of the sort of thing that predicate expressions (predicate letters and $\lambda$-expressions) denote. Predicate expressions denote set of objects, so $\forall$ and $\exists$ denote sets of sets of objects:

- $I_g(\forall) = I(\forall) = \{S \subseteq D : S = D\} = \{D\}$

- $I_g(\exists) = I(\exists) = \{S \subseteq D : S \neq \emptyset\} = \mathcal{P}(D) \setminus \{\emptyset\}$

($\mathcal{P}(D)$ is the powerset of $D$, the set of all its subsets.)
Figure 1: Venn diagram illustrating interpretation of most

We’ve switched from “$|=\,$” to $I$ because we’re talking about the denotation of a function. Let’s try working out an example:

$$(D, I) \models g \exists (\lambda x. Px \land Qx) \iff I_g((\lambda x. Px \land Qx)) \in I_g(\exists)$$

iff $${u : (D, I) \models g[x:=u] \, Px}$$
and $$(D, I) \models g[x:=u] \, Qx} \in \mathcal{P}(D) \setminus \{\emptyset\}$$

iff $${u : I_{g[x:=u]}(x) \in I(P) \land I_{g[x:=u]}(x) \in I(Q)} \neq \emptyset$$
iff $${u : I(P) \land u \in I(Q)} \neq \emptyset$$
iff $$(P) \cap I(Q) \neq \emptyset$$

which is pretty much where we hoped to come out.

Okay, now that we’ve reduced quantifiers to predicates on predicates, one way to generalize is pretty clear: Scale up to binary relations on (unary) predicates. For example (see figure 1)

$$(P_1, P_2) \iff |P_1 \cap P_2| > |P_1 \setminus P_2|$$

or, to be preciser but tiresomer about it,

$$I(Most) = \{(P_1, P_2) : \ldots\}$$

because the denotation of Most is a set of ordered pairs of sets of domain objects.\(^1\)

We have arrived at generalized quantifier theory. Our binary quantifiers are what logicians call quantifiers of type $\langle 1, 1 \rangle$, which suggests we’ve just dipped our toes into a big ocean. However, for purposes of translating natural language into internal representation, it’s seldom necessary to venture any further.

\(^1\)On infinite domains, this condition is harder to state. Fortunately, our textbook is focused [[ like a laserbeam ]] on the finite case. By the way, it’s possible, but definitely not easy, to prove you can’t define Most in terms of $\forall$ and $\exists$; in fact, you can’t do it no matter how how many other one-place quantifiers you introduce into the language. [[ See Kolaitis and Väänänen 1995, ch. 14, thm. 14, cited by Peters and Westerståhl 2006, p. 62. ]]
I said a couple of lectures ago that it’s weird that to define “every” we have to talk about all the objects in the universe. But we don’t have to, if we redefine “every” as a two-place quantifier, with a very simple semantics:

\[ I(\forall_2) = \{(P_1, P_2) : P_1 \subseteq P_2\} \]

Exercise: How is “Some \( P \) are \( Q \)” defined, as a two-argument quantifier (\( \exists_2 \))? 

A Bit of an Appendix

The quantifier “more . . . than . . .” is of type \( \langle 1, 1, 1 \rangle \). \( \text{more}(P, Q, R) \), where \( P, Q, \) and \( R \) are unary predicates, means that the number of objects satisfying \( P \) and \( R \) is greater than the number satisfying \( Q \) and \( R \). “More dogs than cats inherit money” might have the internal representation

\[
\text{more}(\lambda x. \text{species}(x, \text{dog})), \\
(\lambda x. \text{species}(x, \text{cat})), \\
(\lambda x. \exists_2(\lambda m. \text{quantity\_money}(m)), \\
(\lambda y. \text{inherit}(x, y)))
\]

What’s the appropriate internal representation of this example: “More married couples than unmarried couples have children”? You could represent a bit of set theory, and put ordered pairs into your ontology:

\[
\text{more}(\langle \lambda M.M = \{(p_1, p_2) : \text{married}(p_1, p_2)\}\rangle, \\
(\lambda U.U = \{(p_1, p_2) : \text{couple}(p_1, p_2) \land \neg \text{married}(p_1, p_2)\}\rangle), \\
(\lambda C.C = \{(p_1, p_2) : \exists x(\text{child}(x, p_1, p_2))\}\rangle)
\]

But \( \lambda \)-expressions are just as powerful as sets already. A curried \( n \)-argument \( \lambda \)-expression \( p \) of type \( e \to e \to \ldots \to t \) represents the same information as a set of \( n \)-tuples:

\[
\{\langle x_1, \ldots, x_n \rangle : px_1x_2 \ldots x_n = T\}
\]

If you’re willing to entertain the idea of a quantifier of type \( \langle 2, 2, 2 \rangle \), you could analyze our sentence that way:

\[
\text{more}_{\langle 2, 2, 2 \rangle}(\text{married}, \\
(\lambda p_1.\lambda p_2. \text{couple}(p_1, p_2) \land \neg \text{married}(p_1, p_2)), \\
(\lambda p_1.\lambda p_2. \exists x(\text{child}(x, p_1, p_2))))
\]
Of course, the way we analyzed the semantics of \( \lambda \)-expressions involved sets of objects, so all we’ve done is move them from the theory to the meta-theory. But the meta-theory is full of set theory already. We’ve simplified our actual ontology by keeping it out. Plus our \( \lambda \)-expressions are strongly typed (even though we’ve kept the types offstage), so we don’t have to worry about paradoxes.

By the way, what we were calling \( \forall_2 \) and \( \exists_2 \) would now be called \( \forall_{(1,1)} \) and \( \exists_{(1,1)} \).