Notes on generalized quantifiers

Ann Copestake (aac@cl.cam.ac.uk), Computer Lab

These notes are (minimally) adapted from lecture notes on compositional semantics used in a previous course. They are provided in order to supplement the notes from the Introduction to Natural Language Processing module on compositional semantics.

1 More on scope and quantifiers

In the introductory module, we restricted ourselves to first order logic (or subsets of it). For instance, consider:

(1) every big dog barks
$\forall x[\text{big}'(x) \wedge \text{dog}'(x) \Rightarrow \text{bark}'(x)]$

This is first order because all predicates take variables denoting entities as arguments and because the quantifier ($\forall$) and connective ($\wedge$) are part of standard FOL.

As far as possible, computational linguists avoid the use of full higher-order logics. Theorem-proving with anything more than first order logic is problematic in general (though theorem-proving with FOL is only semi-decidable and that there are some ways in which higher-order constructs can be used that don't make things any worse). Higher order sets in naive set theory also give rise to Russell's paradox. Even without considering inference or denotation, higher-order logics cause computational problems: e.g., when deciding whether two higher order expressions are potentially equivalent (higher-order unification).

However, we have to go beyond standard FOPC to represent some constructions. We need *modal operators* such as *probably*. I won't go into the semantics of these in detail, but we can represent them as higher-order predicates. For instance:

$$\text{probably}'(\text{sleep}'(k))$$

corresponds to *kitty probably sleeps*. In fact, rather than using the FOPC symbol for negation, as we've done up to now, we'll use not$'$ in a similar way to the modal operators, for instance:

$$\text{not}'(\text{sleep}'(k))$$

We also have to consider verbs such as *believe* which take propositions as arguments. For example:

$$\text{believe}'(l, \text{not}'(\text{sleep}'(k)))$$

corresponding to *Lynx believes it is not the case that Kitty sleeps* or *Lynx believes Kitty doesn't sleep*.

We also need determiners other than *every* and *some*. We can represent some other determiners (and complex determiners) using just the standard FOPC quantifiers. For instance 'at least 2 As are Bs' could be represented as:

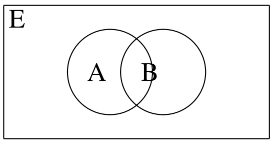
$$\exists x[\exists y[x \neq y \wedge A(x) \wedge A(y) \wedge B(x) \wedge B(y)]]$$

But this sort of analysis is cumbersome and can't be extended to all determiners. For instance, to give a representation of *most*, as in *most As are Bs*, we need to be able to talk about the number of As which are Bs relative to the total number of As. This can't be done in terms of standard FOL. We will represent *most*, and in fact all other quantifiers, using *generalized quantifiers* (GQs). GQs are much more suited to natural language representation than standard FOPC quantifiers.

1.1 An introduction to generalized quantifiers.

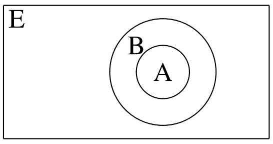
Generalized quantifiers involve the relationship between two sets of individuals, A and B, within a domain of discourse E. Notation: $D_E AB$.

Pictorially:



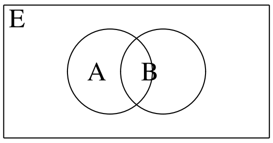
It turns out that for true quantifiers, all we need to know about is the cardinality of the set A and the cardinality of the set $A \cap B$ (i.e., $|A|$ and $|A \cap B|$). (More detail about this is given in §1.3, below.) Using generalized quantifiers it is possible to represent (nearly) all natural language determiners, plus expressions like *at least two* etc.

For instance, *every*:



$|A| = |A \cap B|$

some:



$|A \cap B| \geq 1$

at least two: $|A \cap B| \geq 2$

most (interpreted as *more than half*): $|A \cap B| > |A|/2$

Exercise: Express the meaning of the following determiner phrases in terms of the cardinality of A and B: *no*, *not every*, *all but three*, *few*.

Generalized quantifiers were described by Mostowski (1957), but the first systematic linguistic application is Barwise and Cooper (1981).

1.2 Terminology and notation

The set A is referred to as the *restriction* of the quantifier and B as the *body* (or sometimes (a little confusingly) as the *scope*). For instance, in *every white cat likes Kim*, the Nbar *white cat* corresponds to the restriction of the quantifier and *likes Kim* to the body.

Notationally, we could use lambda expressions for generalized quantifiers.

(2) every white cat likes Kim
$\text{every}'(\lambda x[\text{white}'(x) \wedge \text{cat}'(x)], \lambda y[\text{like}'(y, k)])$

But it is useful to adopt a representation of the form: quantifier(bound-variable,restriction,body). The bound variable is really for notational convenience so we don't have to bother with the lambda variables.

(3) every white cat likes Kim
$\text{every}'(x, \text{white}'(x) \wedge \text{cat}'(x), \text{like}'(x, k))$

Either the restriction or the body of a quantifier may contain another quantifier:

(4) most white cats like some squeaky toys
$\text{most}'(x, \text{white}'(x) \wedge \text{cat}'(x), \text{some}'(y, \text{toy}'(y) \wedge \text{squeaky}'(y), \text{like}'(x, y)))$ (preferred reading)
$\text{some}'(y, \text{toy}'(y) \wedge \text{squeaky}'(y), \text{most}'(x, \text{white}'(x) \wedge \text{cat}'(x), \text{like}'(x, y)))$ (dispreferred reading)

(5) most mothers of two white cats like Kim
$\text{most}'(x, \text{two}'(y, \text{white}'(y) \wedge \text{cat}'(y), \text{mother}'(x, y)), \text{like}'(x, \text{Kim}'))$
$\text{two}'(y, \text{white}'(y) \wedge \text{cat}'(y), \text{most}'(x, \text{mother}'(x, y), \text{like}'(x, \text{Kim}')))$

We will discuss scope ambiguity with generalized quantifiers a bit more below. Note these examples are simplified for discussion purposes. Whether *two* should be represented as a quantifier is a matter of debate. Also proper names like *Kim* don't really correspond to constants.

If we restrict ourselves to the quantifiers *some* and *every*, we can straightforwardly translate any expression using generalized quantifiers into FOL. In the case of *some*, we simply conjoin the restriction and the body, while for *every* we put the restriction on the left hand side of an implication and the body on the right hand side.

1.3 More about quantifiers (optional)

In the following section, I'll go through the properties of true quantifiers, which explain the generalization that only the cardinality of the restriction and the intersection of the restriction and the body are relevant to interpretation. Interestingly enough, natural language determiners are nearly all true quantifiers.

Not all relationships between two sets correspond to quantifiers. True quantifiers obey *extension*, *conservativity* and *isomorphy*.

Extension:

EXT: for all $A, B \subseteq E \subseteq E' : D_E AB \leftrightarrow D_{E'} AB$

This means that the relation stays true or false even if we extend the universe, so the only individuals we need to worry about are those in A or B and we can forget about E.

Consider a world with exactly 85 individuals, a set A consisting of 20 individuals and B with 80 individuals, where 15 members of A are in B. Then 'most A are B' is true. Expand the universe to 1000 individuals maintaining A and B: then 'most A are B' is still true.

Exercise: Does *many* observe EXT if it can mean something like *relatively many*? For instance, consider the scenario above, but in the expanded universe there are also multiple sets C, D, F etc, with more than 60 members which are members of B. Do you think *Many A are B* is true in both the initial and the expanded universe?

Conservativity:

CONS: for all $A, B \subseteq E : D_E AB \leftrightarrow D_E A(A \cap B)$

This means the quantifier 'lives' on its first argument: if we're considering the truth of *all dogs bark* we only need to look at the set of dogs — the set of cats is irrelevant. (An exception to CONS is *only*, but this may well not be a determiner).

If we assume CONS, the truth of DAB only depends on A and $A \cap B$

Isomorphy:

ISOM: If f is a bijection from E to E', then $D_E AB \rightarrow D_{E'} f(A) f(B)$

This means we don't care about the actual elements of the sets A and B, all we care about are the cardinalities of A and $A \cap B$ (i.e., $|A|$ and $|A \cap B|$).

1.4 Generalized quantifiers and generalizations (optional)

The fact that natural language quantifiers seem to obey the constraints described above indicates an interesting property of natural language. Generalized quantifier theory has also made it possible to formalize several other observations about natural language in terms of properties of quantifiers.

For instance, quantifiers are:

Upwardly monotonic if DAB and $B \subseteq B'$ then DAB'

Downwardly monotonic if DAB and $B' \subseteq B$ then DAB'

Only downwardly monotonic quantifiers allow 'negative polarity items' (e.g. *any*, *to lift a finger*) in their scope.

(6) * Every cat ate any of the mice

(7) No cat ate any of the mice

(8) Nobody lifted a finger to help

(9) *Everybody lifted a finger to help

The true story is much more complicated than these examples would indicate. Negative polarity is an important topic in formal semantics: in computational linguistics, it is of little relevance for interpretation, but potentially significant in generation.

1.5 Scope ambiguity expressed with generalized quantifiers

We have already seen scope ambiguity with FOPC but here we will discuss it in more detail. Consider the following:

(10) Every person in the room speaks two languages.

This is standardly assumed to have two readings: the preferred one is that every person is bilingual, the other is that there are two languages every person shares.

(11) every′$(x,$ person′$(x) \wedge$ in-room′$(x),$ two′$(y,$ language′$(y),$ speak′$(x,y)))$ (preferred reading)
two′$(y,$ language′$(y),$ every′$(x,$ person′$(x) \wedge$ in-room′$(x),$ speak′$(x,y)))$ (dispreferred reading)

If you find it difficult to get the dispreferred reading, consider:

(12) Every person in the room speaks two languages. Those languages are French and German.

Similarly:

(13) Everyone approves of two US presidents: Lincoln and FDR. Opinions on the rest are less consistent.

Sometimes scope ambiguity in the logical form does not affect meaning:

(14) Some students attend some lectures
some′$(x,$ student′$(x),$ some′$(y,$ lecture′$(y),$ attend′$(x,y))$
some′$(y,$ lecture′$(y),$ some′$(x,$ student′$(x),$ attend′$(x,y))$

These two forms correspond to exactly the same models.

Supposing we choose to build just one form when we're parsing — this implies the grammar won't accept the other form when we're generating. This is a particular instance of the problem of *logical form equivalence*, which is a problem for language generation.

Often one reading subsumes the other: e.g., in the classic example *every man loves some woman*, the LF corresponding to the preferred reading is valid in all the models that correspond to the single woman reading. Note that this isn't true for *exactly two languages*.

Practically, resolving ambiguity can be very difficult: in fact, it has the potential to be AI-complete because of the effect of context. Also, it is very expensive to get multiple parses, if scope ambiguity is made to correspond to syntactic/lexical ambiguity. Scope ambiguities may not need to be resolved anyway (e.g., they very rarely make a difference in MT between English and German). Most current systems either don't represent scope (but then get into problems if they do try and do inference) or they use an underspecified representation.