This paper introduces some of the main components of a novel type theoretical semantics for quantification with plural noun phrases. This theory, unlike previous ones, sticks to the standard generalized quantifier treatment of singular noun phrases and uses only one lifting operator per semantic category (predicate, quantifier and determiner) for quantification with plurals. Following Bennett (1974), plural individuals are treated as functions of type \( e^t \). Plural nouns and other plural predicates accordingly denote \( (e^t)t \) functions. Such predicates do not match the standard \( (e^t)((e^t)t) \) type of determiners. Following Partee and Rooth (1983), type mismatches are resolved using type shifting operators. These operators derive collectivity with plurals, keeping the analysis of singular noun phrases, where no type mismatch arises, as in Barwise and Cooper (1981). A single type shifting operator for determiners combines into one reading the existential shift and the counting (neutral) shift of Scha (1981) and Van der Does (1993). This operatorcombines the conservativity principle of generalized quantifier theory with Szabolcsi’s (1997) existential quantification over witness sets. The unified lift prevents unmotivated ambiguity as well as the monotonicity ill of existential lifts pointed out by Van Benthem (1986).

Bennett’s typing of plurals is based on the distinction between atomic entities of type \( e \) and sets of atomic entities characterized by \( e^t \) functions. This distinction is reflected in the lexical type of natural language predicates, which are divided into two subclasses:

1. **Atom predicates** of the lexical type \( e^t \), including intransitive verbs like *sleep*, *sing* and *dance* and nouns like *student*, *teacher* and *committee*.\(^1\)

2. **Set predicates** of the lexical type \( (e^t)t \) (abbreviated \( ett \)), including intransitive verbs like *meet*, *gather* and *disperse* and relational nouns like *friend*, *brother* and *colleague*.

Non-lexical predicates, inflected with number agreement features, may be of a different type than their lexical type according to the following rules of thumb.

1. **Morphologically singular** predicates are of type \( e^t \), even when their lexical type is \( ett \).

2. **Morphologically plural** predicates may get the \( ett \) type, even when their lexical type is \( et \).\(^3\)

For example, the singular nouns *student* and *friend* and the singular verbs *sleep* and *meet* are of type \( et \). The plural nouns *students* and *friends* and the plural verbs *sleep* and *meet* are of type \( ett \).

This typing is compositionally derived using the following denotations assumed for the (sometimes covert) number features +SG (singular) and +PL (plural). The +SG feature denotes the function that lowers an \( ett \) predicate to type \( et \) by unioning the singletons in its extension. The +PL

---

\(^1\)In Winter (1998,1999) I argue that the distinction between atom predicates and set predicates should not be the classical distributive/collective typology of predicates and suggest a new empirical test for the atom/set classification.

\(^2\)According to the criterion mentioned in footnote 1, and following Barker (1992) and Schwarzschild (1996), “group denoting” nouns like *committee* and *senate* have the same semantic type as other simple nouns.

\(^3\)In fact, as will be remarked below, plural predicates are assumed to be ambiguous between types \( et \) and \( ett \).
feature denotes the identity function on \( et \) predicates. Formally:

\[
\llbracket +SG \rrbracket = sg_{(et)(et)}^{def} \lambda P_{et}. \lambda x. P(\{x\}) \\
\llbracket +PL \rrbracket = id_{(et)(et)}^{def} \lambda P_{et}. P
\]

Note that the \( +SG \) and \( +PL \) number features denote functions whose arguments are of type \( et \). This creates a type mismatch with atom predicates, which are lexically of type \( et \). Following Partee and Rooth (1983), this type mismatch triggers a process of type fitting. The proposed operator for predicate fitting is defined as follows.

\[
pfil_{(et)(et)}^{def} \lambda P_{et}. \lambda A_{et}. [\_] \neq A \subseteq P
\]

This is the distributivity operator of Link (1983), which also resolves type mismatches in coordinations like \( meet \) and \( sleep \), commonly used to argue against Bennett’s typing.

The following examples illustrate the use of the number feature denotations and the \( pfil \) operator. In (1), the \( pfil \) operator fits the \( et \) type of the atom predicate \( student \) to the type of the singular/plural number feature. In (2), with the set predicate \( meet \), type fitting is unnecessary.

(1) a. \( \llbracket [student] \rrbracket = [student]+SG = student'_{et} \quad sg_{(et)(et)}^{def} \)
   \[
   sg(pfil(student')) = student'
   \]
   (mismatch)

b. \( \llbracket [students] \rrbracket = [student]+PL = student'_{et} \quad id_{(et)(et)}^{def} \)
\[
   id(pfil(student')) = \lambda A_{et}. [\_] \neq A \subseteq student'
   \]
   (mismatch)

(2) a. \( \llbracket [meets] \rrbracket = [meet]+SG = sg_{(et)(et)}^{def}(meet'_{et}) = \lambda x. \lambda x. meet'\llbracket \{x\}\rrbracket \)
   \[
   b. \llbracket [meet] \rrbracket = [meet]+PL = id_{(et)(et)}^{def}(meet'_{et}) = meet'
   \]

Note that the composition of the singularity feature denotation \( sg \) with the \( pfil \) operator gives the identity function on the \( et \) domain, as stated below.

**Fact 1** For every \( P_{et} : sg(pfil(P)) = P \).

This means that in the proposed mechanism singular morphology has no semantic effect on the lexical denotation of atom predicates, just like plural morphology (whose meaning is \( id \)) does not trigger any change in the denotation of set predicates.

Unlike predicates, all lexical determiners are assumed to range over atoms: their type is uniformly the standard \( (et)(ett) \). In a singular sentence like \( no \ student \ sleeps \) we get the following standard analysis.

(3) \( no'_{(et)(et)}(student'_{et})(sleep'_{et}) \)
   \[
   \Leftrightarrow student'_{et} \cap sleep'_{et} = \emptyset
   \]

In the singular sentence \( no \ committee \ meets \), the denotation of the singular set predicate \( meets \) is derived using the \( pfil \) and \( sg \) operators, which leads to the following (plausible) interpretation.

(4) \( no'_{(et)(et)}(committee'_{et})(sg(pfil(meet'_{et}))) \)
\[
\Leftrightarrow committee'_{et} \cap \{x : meet'_{et}(\{x\})\} = \emptyset
\]

In plural sentences like \( no \ students \ slept \) or \( all \ the \ committees \ met \), we get a type mismatch between the \( (et)(ett) \) determiner and the \( ett \) plural noun. This is resolved by combining two different processes of \( counting \) and \( existential \ quantification \). To illustrate the semantic outcome of these processes in full generality, consider the following example.
(5) Exactly five students met.

The exact counting of students who met in (5) is achieved in two steps: (i) a conservativity step, intersecting the denotation of meet with the denotation of students; (ii) a participation step, unioning both sets of sets that serve as arguments of the determiner. This leads to the following analysis.

\[
(6) \text{exactly}_5'(\bigcup p\text{fil}(\text{student}'))(\bigcup \text{meet}' \cap p\text{fil}(\text{student}')) \\
\quad \Leftrightarrow |\{x \in \text{student}' : \exists A \subseteq \text{student}'[x \in A \land \text{meet}'(A)]\}| = 5
\]

In words: the total number of students who participated in student meetings is exactly five.

The counting process is defined in general below as a relation between a determiner and two ett predicates.

**Definition 1** We say that a determiner \(D_{(\text{et})}(\text{et})\) counts the ett predicates \(\mathcal{A}\) and \(\mathcal{B}\), and denote \(\text{count}(D)(\mathcal{A})(\mathcal{B})\), iff \(D(\bigcup \mathcal{A})(\bigcup(\mathcal{A} \cap \mathcal{B}))\) holds.

Counting does not yet make sure a meeting of five students actually took place in sentence (5), as intuitively required. For instance, if the only sets of students in the extension of the predicate meet are the sets \(\{s_1, s_2\}\) and \(\{s_3, s_4, s_5\}\), where \(s_1, \ldots, s_5\) are different students, then (6) is formally true whereas sentence (5) is intuitively false, or highly incoherent. To capture this effect, we add an additional existential process to the count condition. This process is a slightly modified version of the proposal in Szabolcsi (1997) to quantify over witness sets. A set \(W\) is a witness of a determiner \(D\) and a set \(A\) iff \(W \subseteq A\) and \(D(A)(W)\) holds. The witness condition between a determiner \(D_{(\text{et})}(\text{et})\) and two ett predicates is defined as follows.

**Definition 2** We say that a determiner \(D\) witnesses the ett predicates \(\mathcal{A}\) and \(\mathcal{B}\), and denote \(\text{wit}(D)(\mathcal{A})(\mathcal{B})\), iff either \(\mathcal{A} \cap \mathcal{B} = \emptyset\) or there exists \(W \in \mathcal{A} \cap \mathcal{B}\) such that \(D(\bigcup \mathcal{A})(W)\) holds.

In other words: \(D\) witnesses \(\mathcal{A}_{\text{et}}\) and \(\mathcal{B}_{\text{et}}\) iff whenever \(\mathcal{A} \cap \mathcal{B}\) is not empty it includes a witness of \(D\) and \(\bigcup \mathcal{A}\). For (5), the witness condition derives the following requirement.

\[
(7) \text{wit}(\text{exactly}_5'(\text{et})_{(\text{et})})(\text{et})((p\text{fil}(\text{student}'_{\text{et}}))(\text{meet}'_{\text{et}})) \\
\quad \Leftrightarrow \exists A \subseteq \text{student}'[A \neq \emptyset \land \text{meet}'(A)] \rightarrow \exists A \subseteq \text{student}'[|A| = 5 \land \text{meet}'(A)]
\]

In words: if any student(s) met then there was a meeting of exactly five students.

In Scha (1981) and Van der Does (1993) the existential effect is obtained by a separate reading of determiners, in addition to a “neutral” reading that generates analyses as in (6) (cf. footnote 4). This ambiguity strategy suffers from the fact that neither reading is completely adequate to the semantics of sentences like (5). As mentioned above, the counting reading in (6) ignores the implication in (5) that a meeting of exactly five students took place. The existential reading (7) requires the existence of such a meeting, but creates a more severe problem pointed out in Van Benthem (1986:52-53): “existential” readings like (7) actually allow more than five students to participate in meetings in a sentence like (5). In this way, the existential reading counter-intuitively models all quantifiers as upward monotone.

To solve these problems, the wit operator is devised so that it combines well with the counting reading into a unified operator for quantification with plurals. The conjunction of (6) and (7) correctly analyzes plural sentences like (5) without hurting the (non)monotonicity properties of determiners.

---

4This counting process has semantic implications similar to the “neutral” reading of Scha (1981), which is defined in Van der Does (1993) as a relation \(N\) between a determiner, an \(\text{et}\) predicate and another \(\text{ett}\) predicate: \(N(D_{(\text{et})}(\text{ett}))(A_{\text{et}})(B_{\text{ett}}) = \text{count}(D)(p\text{fil}(A))(B)\).
This conjunction yields the following general definition of the determinant fitting operator, mapping atom-based determiners to set-based determiners.

\[ d_{\text{fit}}(\textit{all}(\textit{et}))(\textit{student}(\textit{et})))(\textit{smile}(\textit{et})) \equiv \lambda D. \lambda A. \lambda B. \text{count}(D)(A)(B) \land \text{wit}(D)(A)(B) \]

The \( d_{\text{fit}} \) operator derives an \( \text{ett} \) type quantifier as the denotation of plural noun phrases. This leads to a type mismatch between such quantifiers and \( \text{ett} \) quantifiers, as for example in the following sentence.

(8) All the students and every teacher smiled.

This kind of type mismatch is resolved by a quantifier fitting operator that preserves the distributivity of singular quantifiers and is defined as follows.

\[ q_{\text{fit}}(\textit{all}(\textit{et}))(\textit{student}(\textit{et}))(\textit{smile}(\textit{et})) \equiv \lambda Q. \lambda A. Q(sg(A)) \]

In words: a quantifier \( q_{\text{fit}}(Q) \) holds of the sets of sets whose singleton members’ union is in \( Q \). Using the \( q_{\text{fit}} \) operator, sentence (8) is analyzed as follows.

\[
(9) \quad ((d_{\text{fit}}(\text{all}(\textit{et}))(\text{student}(\textit{et})))) \cap (q_{\text{fit}}(\text{every}(\textit{et}))(\text{teacher}(\textit{et}))))(\text{smile}(\textit{et})) \\
\quad \iff \text{student}(\textit{et}) \subseteq \text{smile}(\textit{et}) \land \text{teacher}(\textit{et}) \subseteq \text{smile}(\textit{et})
\]

Note that \( \text{every} \) and \( \text{all} \) are treated as synonyms, both of them denoting the subset relation between \( \textit{et} \) predicates.

The following facts on the resulting type fitting system show some of its semantic features.

**Fact 2** For all \( D(\textit{et})(\textit{ett}) \), \( A(\textit{et}), B(\textit{et}) \): \( d_{\text{fit}}(D)((p_{\text{fit}}(A))(p_{\text{fit}}(B)) \iff D(A)(A \cap B) \).

This shows that for plural sentences like \textit{all the students slept}, where both the noun and the verb are lexically of type \( \textit{et} \), the truth conditions derived by type fitting are equivalent to the standard ones, due to the conservativity of natural language determiners \( D(A)(B) \iff D(A)(A \cap B) \).

**Fact 3** For all \( D(\textit{et})(\textit{ett}) \) \( \in \text{MON} \), \( A(\textit{ett}), B(\textit{ett}) \): \( d_{\text{fit}}(D)(A)(B) \iff \text{count}(A)(B) \).

This shows that for sentences with right downward monotone determiners, the existential \text{wit} requirement within \( d_{\text{fit}} \) is redundant. For instance, the meaning derived for sentences like \textit{less than five students slept} correctly does not require any existence of a set of students that did something. This solves the Van Benthem problem for naive existential techniques.

**Fact 4** For all \( D(\textit{et})(\textit{ett}) \), \( A(\textit{ett}), B(\textit{ett}) \): \( q_{\text{fit}}(D(A))(p_{\text{fit}}(B)) \iff D(A)(B) \).

This establishes the preserved distributivity of \( \textit{ett} \) quantifiers \( q_{\text{fit}} \)ed to \( \textit{ett} \ttt \).

**Remarks:**

1. **Conservativity** is preserved with collective quantification: \( D \text{Ns V} \) is equivalent to \( D \text{Ns are Ns that V} \) also when N is a noun like \textit{friend} or V is a verb like \textit{meet}. The conservativity step within \( d_{\text{fit}} \) captures this fact. Lexical **monotonicity** of determiners is however not always preserved when they combine with \( \textit{ett} \) predicates. For instance, \textit{all the students drank together a whole glass of beer} does not entail \textit{all the rich students drank together a whole glass of beer}, in contrast to the left downward monotone behavior of \( \textit{all} \) with \( \textit{et} \) predicates. The \textit{wit} part of \( d_{\text{fit}} \) captures this contrast, requiring that in the antecedent there was a set of students (not necessarily rich!) that drank together a whole glass of beer. This correctly cancels the \( \downarrow \text{MON} \) entailment.
2. There is a need to allow plural predicates to get also the \textit{et} type meaning of their singular form. For instance, the sentence \textit{all the committees met} is ambiguous between two readings. One reading requires that there was a joint meeting of all the committees together. Another reading states that every committee had (its own) meeting. The first reading is already captured by the \textit{d fil} operator, whose application is driven by the \textit{ett} type of the plural noun \textit{committees}. This type of plural nouns is in turn driven by the denotation of the plurality feature \textit{+PL}. The second reading can be obtained by stipulating that the \textit{+PL} feature is ambiguous between the \textit{id}_{\text{ett}}(\text{ett}) meaning assumed above and the \textit{sg}_{\text{ett}}(\text{ett}) meaning of the \textit{+SG} feature. Using the latter reading, all plural sentences can also get the standard uniform analysis of quantifiers in Barwise and Cooper (1981), in addition to their lifted meaning as derived in this paper.

References


Computer Science
Technion
Haifa 32000
Israel
E-mail: winter@cs.technion.ac.il
WWW: http://www.cs.technion.ac.il/~winter