A type free theory and collective/distributive predication

the Journal of Logic, Language and Information 4, 1995

Fairouz Kamareddine [*]
University of Glasgow
Department of Computing Science
17 Lilybank Gardens
Glasgow G12 8QQ
Scotland

November 30, 1996

Abstract

The purpose of this paper is to provide a simple type-free set theory which can be used to give the various readings of collective/distributive sentences.

Keywords: Type free set theory, singletons, collectivity, distributivity.

1 Introduction

Consider $f(a)$ where f is a function and a is an argument. There are many cases that influence the reading of $f(a)$. Those cases are 1.1, 1.2 and 1.3 below:

Case 1.1 The function f can be *collective* as in C) where it is applied to the collection *John and Mary*, or *distributive* as in D) where it is applied to each element of the collection of children.

- C) John and Mary *love each other.*
- D) The children are *happy.*

Case 1.2 The function f can be *singular* as in S) where *lazy* is singular — *the dogs are lazy* here means that every dog is lazy — or *plural* as in P) or *mixed* as in M).

- S) John is *lazy.*
- P) Bill and Helen *suit each other.*

---

[*] Part of this work was carried out while the author was on a study leave for one year at the technical university of Eindhoven, the Netherlands. I am grateful to the university of Eindhoven for their hospitality, financial and academic support. Moreover, I am grateful to the anonymous referees for their useful suggestions and comments. Last but not least I would like to thank Kieran Clenanghan for having commented on the style of the paper and Gerry Seligman for having drawn my attention to the problem faced by complex predicates.

- M) The children *carried the piano* or John *carried the piano.*

Case 1.3 The argument a can be either a *single* entity as in OE), a *collection* of entities as in OP), or a *combination* of single entities and collections as in OM).

- OE) *John* is happy.
- OP) *John and Mary* are happy.
- OM) *Mary and the boys* danced.

Hence, the reading of $f(a)$ can be either collective or distributive depending on the nature of the *function* f and of the *argument* a. Moreover the same sentence may have various readings. The following examples will illustrate the point:

Example 1.4 Take the sentence: *The girls ate a pie.* Here the verb *ate a pie* is being applied to the object *the girls.* To interpret a sentence one must be able to give all the possible readings of the sentence. The above sentence can have the different readings in 1 and 2 below:

1. *Each girl* ate a pie.
2. *The girls* ate a pie between them.

Example 1.5 Now consider the sentence: *The girls and the boys danced and sang loudly*, which can have the following interpretations:

1. *The girls* danced and sang loudly and *the boys* danced and sang loudly.
2. *Each of the girls* danced and sang loudly and *each of the boys* danced and sang loudly.
3. *The girls and the boys together* danced and sang loudly.

Example 1.6 For the sentence: *The children built the raft*, we have the following readings:

1. *Each child* built the raft.
2. *All the children together* built the raft.

Example 1.7 Some arguments might be indivisible and so we have to accommodate this fact, for example the sentence: *The crowd gathered in the square.* Here we have an object *the crowd* which is treated as though it were indivisible.

Given all the above, we see that sentences which contain singular/plural functions and collective/distributive readings are difficult to interpret in a unified theory. This may force us into providing quite complex or nonstraightforward theories in the aim of accommodating as much of the above sentences as possible. One should aim however for the following goal:

Goal 1.8 A theory should give for each sentence all and only those readings that are possible for the sentence, while remaining simple and tidy.

The present paper aims at fulfilling Goal 1.8. We will be modest in our aim in that we only attempt extensional examples. The theory however is basically intensional and so could be used for intensional examples. Discussing intensionality in this paper however, will obscure its basic aim of using the set theory that we build to providing the possible readings of sentences. We claim that Concepts 1.9 and 1.10 below, should be used to reach our goal:

Concept 1.9 *(Type Freeness and Set Theory)*
Set theory is simple, tidy and can achieve Goal 1.8 if it was well chosen. A well chosen set theory is a set theory which allows one to treat collections and individuals in the same way. This is because some functions might apply to singular as well as plural entities (*carried the piano* is such a function, where one can say *the boys carried the piano* or *John carried the piano*). One of the properties of such a theory is that it should be type free so that to enable one to treat plurals and singulars in the same way and hence to allow a function to take either individuals or plurals as arguments. In our framework, we will be using a type free set theory where sets will enable us to characterise plurals. Individuals on the other hand, will be a special kind of sets (see Concept 1.10).

Concept 1.10 *(Singletons)*
A type-free theory alone is not enough to give a unified treatment of collectivity and distributivity. In fact, weak versions of such a theory have already been offered by, among others, Link; but the framework is not general enough to allow the representation of all the above cases in a unique framework. What should be aimed at moreover, is a unified framework where collectivity and distributivity can be accommodated and where all possible readings of a sentence can be obtained. For this, we shall need *singletons* which contain themselves as members and which characterize individuals. Treating singletons in this way goes back to [Scha 1981]. We claim however that our theory is more up to date with the requirements of Natural Language Semantics: full expressiveness of λ-calculus, logic and self-reference.

There has been different methods and models employed to represent the different readings of sentences. The two approaches offered in [Link 1983] ... [Link 1986] and [Landman 1989] are particularly important. Landman for example provides a theory where collections, have richer structures than the old sense of (extensional) set theory and Link opened the area of plurals to lattice and group theories. Link's approach however was criticised by Landman and we find Landman's approach still has many shortages. In particular, his system is typed and his models are not very clear. Because he used typed theories, he had to use Partee's type shifting devices to provide the distinction between collective and distributive readings.

This paper attempts to provide a unified way of dealing with examples of functional application where the function can be C), D), S), P) or M) and the argument can be OE), OP) or OM). What enables such a unified framework to be carried out is the use of a *type-free set theory* where a singleton is identified with its member. This theory will enable us to prove some properties of $\cup$, $+$ and * such as: $*f(E+E') = f(E) \wedge f(E')$, $*f(E \cup E') = *f(E) \wedge *f(E')$ and for I_n individual, $*f(I_n) = f(In)$. With such properties, we can provide two tables which summarize the possible readings of sentences depending on the state of the verb. A verb f can be singular or mixed ($*f$ will always be its plural) and collective or distributive. The decision on the status of the verb will always be taken at the start. For example, the verb *lazy* is singular collective whereas the verb *ate* is mixed collective. All these are decisions that are semantic rather than syntactic.

Both the theory and the tables are novel and deserve attention. This novelty is explained in detail in the relevant sections.

2 The type-free Set Theory

2.1 The basic theory $\mathcal{T}_s$

We let E, E', E'', range over Exp, the domain of expressions of $\mathcal{T}_s$ and f range over Fun the domain of functions.

Definition 2.1 *(Syntax)*
The syntax of the language is as follows:
$E ::= T \mid \bot \mid x \mid c \mid \lambda f \mid E'E'' \mid f(E) \mid \Delta E' \mid E' \vee E'' \mid E' = E'' \mid E' \Rightarrow E'' \mid \forall x.E'$
$f ::= basic \mid \uparrow E \mid x.E$
$basic ::= run \mid play \mid dance \mid carry - the\, piano \mid \ldots$

The first clause says what are the expressions, they are Truth (T), Falsity ($\bot$), variables (x), constants (c), the nominals of functions (λf), the aplication of a term to another term ($E'E''$), the application of a function to a term ($f(E')$), the testing of whether an expression is a proposition ($\Delta E'$), the disjunction of two expressions ($E' \vee E''$), the testing of expression equality ($E' = E''$), the implication of one expression to another ($E' \Rightarrow E''$), and the universal quantification ($\forall x.E$).

Functions are basic such as *run, dance*, or are the lifting of an expression into a function as in $\uparrow E$ or the abstraction of an expression in terms of a variable as in $x.E$.

Definition 2.2 *(More Logical Operators)*
Other logical operators are defined in terms of $\wedge, \Rightarrow$ and $\forall$ in the obvious way as follows:
$\neg E =_{df} E \Rightarrow \bot$
$E' \wedge E'' =_{df} (E' \Rightarrow (E'' \Rightarrow \bot)) \Rightarrow \bot$
$\exists x.E =_{df} \neg \forall x.\neg E$
$E' \Leftrightarrow E'' =_{df} (E' \Rightarrow E'') \wedge (E'' \Rightarrow E')$

Definition 2.3 *(Numerals)*
We define $N \subset Exp$ to be the set of numerals defined using any of the known number systems of the type free λ-calculus. For example, we may choose the Church numerals defined as follows:
$0 =_{df} \lambda xy.y$
$1 =_{df} \lambda xy.xy$
$\ldots$
$n =_{df} \lambda xy.x^n y$
$\ldots$

It may be questioned here why we chose such a theory instead of say the type free lambda calculus. The reason for this is that the type free lambda calculus alone is not enough to represent natural language. We need logic too. Logic however is difficult to build on the top of the type free lambda calculus without losing consistency. [Kamareddine 1992A] is a good introduction to the problem of adding logic to the type free lambda calculus. Moreover, someone might point out that there is an obvious way of building propositional connectives on the top of the type free lambda calculus without problems. The point however is that quantifiers are needed in the interpretation of sentences of natural language (see [Kamareddine 1992B] and [Kamareddine 1992C]). Even if we were to concentrate on singulars and plurals, and forget about the interpretation of a bigger subset of natural language, quantifiers will still be needed. In Definition 2.8, for example, the quantifier is present in the definition of the plural function. [Kamareddine, Klein 1993] gives an interpretation of a big fragment of natural language in a type free λ-calculus with logic. The importance of quantification is highly illustrated there. [Kamareddine 1992D] shows that the availability of type freeness with logic enables the representation and typechecking of self referential terms.

Now that the type free lambda calculus alone is not enough, we try to extend it by adding to the syntax of terms, all those terms which involve logic. Hence our terms, T, $\bot$, $E' \wedge E''$, $\forall x.E'$ and so on, which are present in the BNF syntax of terms in Definition 2.1 are obvious. The only non obvious term is $\Delta E'$. Also this is explained in detail in [Kamareddine 1992A]. Here however, we summarize its importance by saying that the paradox is avoided in this theory by the fact that not everything can be a proposition. In particular obtaining something of the form $a = \neg a$ is not contradictory. It will only be contradictory if one can show that Δa. This will get more obvious when one looks at the axioms and rules which are listed below.

2.2 The Axioms

For the axioms, we have the following

$$\uparrow \lambda f = f \tag{1}$$

$$(\lambda f)E = f(E) \tag{2}$$

$$(x.E)E' = E[E'/x] \tag{3}$$

$$\Delta T, \Delta\bot, \Delta(E' = E'') \tag{4}$$

$$\frac{E}{\Delta E} \tag{5}$$

$$\frac{\Delta E' \qquad \Delta E''}{\Delta(E' \vee E'')} \tag{6}$$

$$\frac{E' \qquad \Delta E''}{E' \vee E''} \qquad \frac{E'' \qquad \Delta E'}{E' \vee E''} \tag{7}$$

$$\frac{E' \vee E'' \qquad \begin{matrix}\{E'\}\\ \vdots \\ E\end{matrix} \qquad \begin{matrix}\{E''\}\\ \vdots \\ E\end{matrix}}{E} \tag{8}$$

$$\frac{\Delta E' \qquad \Delta E''}{\Delta(E' \Rightarrow E'')} \tag{9}$$

$$\frac{\Delta E' \qquad\qquad \begin{array}{c}\{E'\}\\ \vdots\\ E''\end{array}}{E' \Rightarrow E''} \tag{10}$$

$$\frac{E' \qquad\qquad E' \Rightarrow E''}{E''} \tag{11}$$

$$\frac{\perp}{E} \tag{12}$$

$$\frac{Ex}{\forall x.E} \qquad x \text{ not free in } E \text{ or any open assumption} \tag{13}$$

$$\frac{\forall x.E}{EE'} \tag{14}$$

$$\frac{\Delta(Ex)}{\Delta(\forall x.E)} \qquad x \text{ not free in } E \text{ or any open assumption} \tag{15}$$

$$\frac{E = E' \qquad\qquad E}{E'} \tag{16}$$

Axiom (1) is obvious, it says that if we look for the function part of the nominalisation of f, we get back f. Axiom (2) says that applying the nominalisation of f to E gives the same result as applying f to E. Axiom (3) is the well known β-conversion principle of the λ-calculus, which comes unrestricted because we assume the substitution takes care of variable clashes. Axioms (4) ... (11) are the logical axioms of the theory, which are explained in detail in [Kamareddine 1988A].

Again, the consistency of such a theory is discussed in [Kamareddine 1988A]. It is worth here however, repeating briefly how this theory avoids, say, Russell's paradox. The sentence $R = \lambda x.\neg xx$ is well formed according to our syntax.

$RR = (\lambda x.\neg xx)(R)$ from Axiom 2
$= (\neg xx)[R/x]$ from Axiom 3
$= \neg RR$ from substitution.

This is not a problem however because RR is any term and there is no possibility of proving ΔR (try to do, you will never get anywhere). In fact, one can state this by saying that, as we are working with the type free λ-calculus, every term E will have a fixed point a (i.e. a point such that $Ea = a$). In particular the term $\lambda x.\neg x$ will have a fixed point a such that $(\lambda x.\neg x)a = a$. That is $a = \neg a$.

This is not contradictory. Just that an object a is equal to its negation. If this object a can be shown to be a proposition, that is, if we can show Δa then we are in trouble. But from the axioms put above, it is not possible to show Δa.

2.3 Sets

As it is obvious so far, we have built logic on the top of the type free λ-calculus. We are however interested in a type free set theory, as sets will play an important role in the interpretation of singular/plural entities. The following definitions aim at introducing this set theory.

Definition 2.4 *(Set Elementhood versus Functional Application)*
We define $\in: Exp \times Exp \to Exp$*, so that:*
For all $E', E'' \in Exp$*, we have:* $(E' \in E'') = E''E'$

Definition 2.5 *(Set Definition)*
We define $D : Exp \to Exp$*, so that:*
For all $E \in Exp$*, we have:* $DE = \forall x.\Delta(x \in E)$

Definition 2.6 *(Set Union)*
We define $\cup : Exp \times Exp \to Exp$*, so that:*
For all $E', E'' \in Exp$*, we have:* $E' \cup E'' = \lambda x.(x \in E' \vee x \in E'')$*.*

Definition 2.7 *(Set of Two Elements)*
We define $+ : Exp \times Exp \to Exp$*, so that:*
For all $E', E'' \in Exp$*, we have:* $E' + E'' = \lambda x.(x = E' \vee x = E'')$*.*

Definition 2.8 *(Plural)*
We define $* : Fun \to Fun$ *such that:*
For all $f \in Fun$*, we have:* $*f = y.\forall x(x \in y \Rightarrow f(x))$*.*

Definition 2.4 is there to relate sets to functions, and a consequence of it is the formula: $(a \in \lambda f) = f(a)$. For example, $x.\neg(xx)$ is a function and $\lambda x.\neg(xx)$ is an object such that $(a \in \lambda x.\neg(xx)) = (x.\neg(xx))a = \neg(aa)$.

Definition 2.5 is there to test the sethood of an expression E, so that if $x \in E$ is a proposition for any x, then E must be a set. For example, any λf where f is a propositional function (i.e. $\Delta f(x)$ is provable for every x) is a set, because $(x \in \lambda f)$ which is $f(x)$ is a proposition for any x and therefore $\Delta(x \in \lambda f)$ is true. As another example, $\lambda x.\neg(x = x)$ is a set.

Definitions 2.4 and 2.5 are the basics for saying what sets are and how we recover them from the theory of functions provided. What is still needed however, are the ways of combining sets together to obtain new sets. This will be very important for the theory of plurals/singulars we are proposing. Singulars will be a special kind of sets (singletons) and plurals will be normal sets. We can combine singulars and plurals using $+$ and $\cup$. Hence, if E' and E'' are sets, then Definition 2.6 forms their union. $+$ forms the set which contains the arguments of $+$ as its only two elements and Definition 2.8 finds the plural function.

All the above definitions are needed to interpret plurals and singulars in our theory. There is also a list of results about these sets that we will need in the following sections (especially Sections 5 and 6). The following lemma states the closure conditions on sets:

Lemma 2.9 *The following holds:*

1. $(E \in \lambda x.E') \wedge (E \in \lambda x.E'') = E \in \lambda x.(E' \wedge E'')$
2. $E \in \lambda x.\neg E' = \neg(E \in \lambda x.E')$
3. $(E \in \lambda x.E') \vee (E \in \lambda x.E'') = E \in \lambda x.(E' \vee E'')$
4. $E \in (E' \cup E'') = (E \in E') \vee (E \in E'')$
5. *If* DE *and* DE' *then* $D(E \cup E')$

6. $D(E+E')$

Proof: *Easy.* □

Remark 2.10 Note that someone might suggest that the definitions of + and * are complex and hence might provide the following definitions:

$a+a=a$ and $a+b=\{a,b\}$ whenever $a \neq b$

$*f(a)=f(a)$ for a an individual and $*f(a) \Leftrightarrow \forall x \in a(f(x))$ when a is a set.

These definitions of + and * are in the meta-language. Our definitions are in the object language. Although intuitively they are equivalent, our approach is better. After all, ZF does not introduce $\cup$ informally. It rather provides axioms which define the various set constructors internally.

The following notion about singleton is similar to (but independent of) the idea of [Scha 1981] of making individuals have themselves as members.

Definition 2.11 *(Singleton)*
We say that a set E is singleton and has label n iff $\downarrow(E,n)$ is true where:
We define $\downarrow$: $Exp \times N \rightarrow Exp$ such that:
For all $E \in Exp$, we have: $\downarrow(E,n)=(E=\lambda x.(x=n \vee x=E))$.

Axiom (17) below defines some singletons. The tags are there to enable us to show the distinctness of these sets. Axiom (18) is needed for the proof of Lemma 2.13 below.

$$\downarrow(John,1) \qquad \downarrow(Mary,2) \qquad \downarrow(peter,3) \qquad \ldots \tag{17}$$

$$f(n) \text{ for every basic function } f \text{ and natural number } n. \tag{18}$$

From Axiom (17) above, John, Mary, are solutions for each of the equations $In=\{n,In\}$. The tag n is introduced to make sure that for $n \neq m$, $In \neq Im$. Each individual such as John, Mary, ... will be interpreted by one of the In. Hence with this representation, individuals are numbered and each individual contains itself. For instance John = {1, John} and John $\in$ John. This means that we can only have a denumerably infinite number of singular individuals. One of our axioms above, axiom (18) states that each basic function holds of n. This restriction enables one to prove Lemma 2.13 below which is needed now that we are treating plurals and singulars in the same manner.

The following lemmas and observations are needed in our work below. They are listed without proofs, as the proofs are easy exercises.

Lemma 2.12 *$*f(E+E')=f(E) \wedge f(E')$ and $*f(E \cup E')=*f(E) \wedge *f(E')$.* □

Lemma 2.13 *For every individual In and every unary function f, $*f(In) \equiv f(In)$.* □

Remark 2.14 *(Removing the tags)* Our addition of the tags above was in order to be able to have distinct individuals and by this addition, Axiom (18) was forced upon us to enable us prove Lemma 2.13. The presence of Axiom (18) might bring out some questions as to whether we will face problems with complex predicates. For example, if *ate a cake* is represented by *ate* then call *n-ate* the function $x.\neg ate(x)$ and if ate(anne) is true but ate(John) is not. Then although ate(Mary) and *ate(Mary) are equivalent, n-ate(John) and *n-ate(John) are not. The reason for this is that n-ate is not basic and so n-ate(1) is not true. In fact, we even have that n-ate(1) = ¬ate(1) is false. This of course might be a problem but it can be avoided by replacing Axioms (17) and (18) by the following two axioms:

$$\downarrow(John) \qquad\qquad \downarrow(Mary) \qquad\qquad \dots \tag{19}$$

$$x \neq x' \text{ for each distinct } x, x' \text{ where } \downarrow(x) \text{ and } \downarrow(x'). \tag{20}$$

and refining $\downarrow$ as $\downarrow(E) = (E = \lambda x.(x = E))$.

Hence, John = { John} and so on. With this change, the above problem is avoided. In this paper, Axioms (17) and (18) will be assumed and not axioms (19) and (20). The whole following sections will still apply however if we assumed axioms (19) and (20) instead.

3 Comparing this type-free set theory to others

3.1 Comparison with ZF

Here, sets are just certain expressions and the reader may want to know which axioms of ZF are true of these objects. We won't go through all the axioms of ZF, but will only concentrate on three.

First the Foundation axiom (which is not really a part of ZF but has been assumed in the tradition of the iterative conception of set) does not hold here. On the contrary, we have:

Type freeness

In this theory, sets can contain themselves.

Example 3.1 The set $\lambda x.(x = x)$ contains itself because: $(\lambda x.(x = x)) \in (\lambda x.(x = x))$.

Second, the axiom of extensionality does not hold here. Rather we are working in an:

Intentional framework

$\forall x(x \in E \Leftrightarrow x \in E')$ does not imply $E = E'$.

Third, the power set axiom does not hold here. To show this we need the following definition which forms the subcollections of a set E:

Definition 3.2 *(Collection of all subcollections)*
We define $\Sigma : Exp \to Exp$, so that:
For all $E \in Exp$, we have: $\Sigma E = \lambda x.\forall y(y \in x \Rightarrow y \in E)$.

The following lemma holds for Σ:

Lemma 3.3 *The following holds:*

1. $(E \in \Sigma E') = \forall y.(y \in E \Rightarrow y \in E')$
2. *If DE and DE' then $\Delta(E' \in \Sigma E)$.*

Proof: *Easy.* □

Remark 3.4 It is not the case however that from DE we can deduce $D(\Sigma E)$. That is, If E is a set, then there is no obvious restriction we can make on the elements of E to obtain ΣE a set.

Note that * and Σ are related, as the following lemma shows:

Lemma 3.5 $\lambda * f = \Sigma\lambda f$.
Proof: $\lambda * f =_{df}$
$\lambda y.\forall x(x \in y \Rightarrow f(x)) =^{Axiom(\ 2)}$
$\lambda y.\forall x(x \in y \Rightarrow (\lambda f)x) =^{Definition(\ 2.4)}$
$\lambda y.\forall x(x \in y \Rightarrow x \in \lambda f) =^{Definition(\ 3.2)}$
$\Sigma(\lambda f)$. □

Remark 3.4 can be now extended to *, due to the relationship between * and Σ. This will lead us to the following Lemma:

Lemma 3.6 *If f is a propositional function then it is not necessarily true that $*f$ is a propositional function. However, if f is a propositional function and if E is a set then $*f(E)$ is a proposition.* □

Example 3.7 $*play = \uparrow (\Sigma\lambda play)$. However from DE we cannot deduce that $D(\Sigma E)$. So we know that $\lambda play$ is a set (as *play* is a propositional function). It is not obvious however if $\Sigma(\lambda play)$ is a set.

3.2 Comparison with some other type free theories with logic

In the past few years, my aim has been to provide *useful* theories whose use might be extended in future work. All these theories aim at combining as much expressiveness and logic. For example, the system of [Kamareddine 1992A] was used to study determiners and quantifiers in that paper. A subsystem of [Kamareddine 1992A] was shown useful in extending Milner's ML with fixed point operators (see [Kamareddine 1992D]) and another subsystem of [Kamareddine 1992A] was shown useful for the semantics of an extended fragment of Montague (see [Kamareddine, Klein 1993]). The fact that these two subsystems are subsystems of [Kamareddine, Klein 1993] is shown in [Kamareddine 199-]. The present theory has of course many common grounds with other theories that aim to have as much expressiveness and logic as possible without getting any paradoxes. The common features between this theory and other ones should not be surprising however. After all, all of them must have the usual λ-calculus axioms and the usual logical axioms. The theory presented in this paper is new however in that it has two levels of basic terms: those in E and those in f. The theory of [Kamareddine, Klein 1993] was typed, so we should say $\lambda x : \sigma.\alpha$ (see that article page 182). This theory is untyped. The theory of [Kamareddine, Klein 1993] avoided the paradox by its types (especially the notion of wb-type on page 184 and the axiom (λ) on page 189, see 2.4 on page 192). The present theory avoids the paradox via the symbol Δ. All in all, the two theories are really different and it would be nice to interpret one in the other, but this is not the subject of the present article. For the present article, I think it would be messy to work with types and to mix functions and objects.

The theory of the present paper is also different from that presented in [Kamareddine 1992A] in that there are two levels of expressions: functions and objects. The present theory has $\uparrow$ and λ which work between functions and terms and the axioms which relate them such as $\uparrow \lambda f = f$, $(\lambda f)E = f(E)$ and $(x.E)E' = E[E'/x]$. For the article of [Kamareddine, Klein 1993], F_0 is enough for the interpretation. Here, I find it important to have both sets which represent functions and individuals.

Moreover, the present theory has the usual set theoretic operators together with the extra ones necessary for plurals and singulars (+,*) and $\downarrow$ together with some results that are new (such as the relation between * and Σ, $\cup$, $+$). As mentioned earlier, the inclusion of $+$, *, $\downarrow$ and so on in this theory was triggered by plurals/singulars. The system presented in this paper is different from all the three systems mentioned above. It has two kinds of basic terms: expressions and functions. Even though all these systems are really modelled by Frege structures, it must be noted that there are various Frege structures depending on the construction followed. Furthermore, various programming languages are given the denotational semantics in a Scott domain. This does not mean that these languages are the same.

4 Semantics

Let us look back at the theory $\mathcal{T}_s$. We see that we have expressions which are constructed using either the construction laws of the type free λ-calculus, $x, c, E'E'', \lambda f$, or using logic[1], $E' \vee E', E' = E'', E' \Rightarrow E'', \forall x.E'$. Of course one cannot question T and $\perp$, they are two particular constants which stand for truth and falsity respectively. $f(E)$ is also obvious. Note too that we departed slightly from the traditional way of constructing terms of the type free λ-calculus. That is, we don't use $\lambda x.E$ where E is an expression open in x, rather, we take any unary function f and map it into an object λf via the operator λ. In fact our operators λ and $\uparrow$ will play a similar role to that of the two embedding functions of the domains of the type free λ-calculus. That is, functions can be mapped to objects, objects can be mapped to functions and there is a certain relation preserved. For example, $\uparrow \lambda f = f$.

Models of the type free λ-calculus however, could not deal with logic added on top of the λ-calculus, since once logic is added, consistency might be threatened (see [Kamareddine 1988A]). For this, there should be some ways of constructing logic on the top of these models of the type free λ-calculus, while remaining consistent. Such a construction was not obvious for a long time. It was initiated by Scott in [Scott 1975] yet the work was incomplete and hence such a model was not achieved. Then came the construction of Frege structures in [Aczel 1980], where simply the idea is to start from any model of the λ-calculus and build logic on top by inductively constructing two collections (the possible propositions and the possible truths) and taking the limit of these two collections which actually draw the logic on the top of the initially considered model of the λ-calculus. As it sounds, the process is quite simple, yet it depends on having a clear idea of the structure and on proving some theorems which will ensure the existence of the various logical connectives in the model considered. Now that logic has been constructed on the top of a model of the λ-calculus, we can consider the structure only in terms of its objects and functions. The objects include propositions and truths and

---

[1] Remember again that only $\Delta E'$ is odd but its reasons were explained earlier.

the functions obey the condition that propositional functions can be projected in the domain of objects (i.e. as sets). Those sets can be applied to any object (hence we now have not only functional application such as $f(x)$ but also the application of one object to another as in $\mathbf{app}(a, b)$), and set application to an object results in a proposition. This is the simple idea of a Frege structure.

Such a model will be what is required for our theory. We need as is obvious, a model which is a model of the type free λ-calculus, which has logic, where sets can contain themselves and where the axiom of extensionality does not hold.

Let us start by more formally introducing Frege structures and show that a Frege structure is actually a model of $\mathcal{T}_s$.

4.1 Frege Structures

We begin with a collection $\mathcal{F}_0$ of objects, and for each natural number $n \geq 1$, we define $\mathcal{F}_n$ as $\{\mathbf{f} : \mathcal{F}_0^n \longrightarrow \mathcal{F}_0\}$, where $\mathcal{F}_0^n = \overbrace{\mathcal{F}_0 \times \mathcal{F}_0 \times \cdots \mathcal{F}_0}^{n\ times}$. In particular, $\mathcal{F}_1$ is the set of all unary functions from $\mathcal{F}_0$ to $\mathcal{F}_0$.

Remark 4.1 For the purpose of our paper, and as we are using unary functions only, it is enough to consider $\mathcal{F}_0$ and $\mathcal{F}_1$. It should be noted however that our choice of unary functions was only to avoid messy details and this is in no way a shortcoming of the paper. Methods of reducing n-ary functions to unary ones are well known.

Within $\mathcal{F}_0$ we pick out $PROP$, the collection of propositions, and $TRUTH$, the collection of all true propositions. (Thus, Aczel makes a crucial departure from Frege in denying that all true propositions can be identified with the True).

So far, then our Frege structures contain objects, functions, propositions and truths. To these, we need to add functionals, logical connectives, and some closure conditions. Aczel proves a theorem however which states that for any number m given, we can find m functionals on the Frege structure (which are primitive and independent, but if these terms are unclear one can forget about them for the purpose of this paper).

The first two functionals that are required in order to provide a model for the lambda calculus and that we single out of that list of functionals which was proved to exist by Aczel, are the following:[2]

$$\lambda : F_1 \longrightarrow F_0$$
$$\mathbf{app} : F_0 \times F_0 \longrightarrow F_0$$

These obey a *comprehension* principle such that whenever $\mathbf{f}$ is a propositional function in $\mathcal{F}_1$, i.e., $\mathbf{f}$ is an element of $\mathcal{F}_1$ which maps its arguments into $\mathbf{PROP}$, then

$$\mathbf{app}(\lambda \mathbf{x}.\mathbf{f}(\mathbf{x}), \mathbf{a}) = \mathbf{f}(\mathbf{a}).$$

λ will be used in interpreting the terms λf of $\mathcal{T}_s$, $\mathbf{app}$ will be used in interpreting the term $E'E''$. Note that the comprehension principle validates Axiom (2).

---

[2] We adopt the notational convention of using boldface terms to denote elements of the model, reserving italics for expressions of the object language. For example, λ is a functional in the model which corresponds to the functor λ in the language.

Before moving to the other functionals, we let $\mathcal{PF}$ be the collection of unary propositional functions in a Frege structure:

$$\mathcal{PF} = \{\mathbf{f} \in \mathcal{F}_1 : \text{for all } \mathbf{x} \text{ in } \mathcal{F}_0, \mathbf{f}(\mathbf{x}) \text{ is in } \mathbf{PROP}\}$$

We can now identify a further subcollection of $\mathcal{F}_0$, namely **SET**, as the individual correlates of propositional functions under λ:

Definition 4.2 *(Sets)*
An object is in **SET** *iff it is* $\lambda\mathbf{f}$ *for some* $\mathbf{f}$ *in* $\mathcal{PF}$.

The distinguishing characteristic of sets (i.e. elements of **SET**) is that they can be predicated of any object in $\mathcal{F}_0$ to yield a proposition:

Definition 4.3 *(Predication)*
If a *is in* **SET**, *then* $\mathbf{app}(a, b)$ *is in* **PROP**, *for any object* b *in* $\mathcal{F}_0$.

Remark 4.4 Note that **SET** in the Frege structure will model the sets we talked about in Section 2.

Comprehension can be restated as follows:

Definition 4.5 *(Comprehension)*
If $\mathbf{f}$ *is in* $\mathcal{PF}$, *then* $\lambda\mathbf{f}$ *is a set* a *such that for any object* b, $\mathbf{app}(a, b)$ *is in* **PROP**, *and* $\mathbf{app}(a, b)$ *is in* **TRUTH** *iff* $f(b)$ *is in* **TRUTH**.

The notions that we have introduced so far — objects in $\mathcal{F}_0$, functions in $\mathcal{F}_1$, **PROP**, **TRUTH**, **SET**, comprehension and predication — are based on a model of the λ-calculus. $\mathcal{F}_0$ will be used in interpreting the expressions of $\mathcal{T}_s$, $\mathcal{F}_1$ will be used in interpreting the functions of $\mathcal{T}_s$. λ will be used to interpret the λ of $\mathcal{T}_s$. What do we use for $\uparrow$? For this we need a way to talk about elements of a Frege structure and will use wffs to do that. We start by mentioning the full abstraction principle of a Frege structure:

Definition 4.6 *(Full abstraction Principle of a Frege structure)*
If $\xi[x_1, x_2, \ldots, x_n]$ *is an expression formed out of objects, functions, functionals and variables (ranging over objects in a Frege structure), where* $x_1, x_2, \ldots, x_n$ *are free variables of* ξ, *then there is a function* f *in the Frege structure such that* $f(a_1, a_2, \ldots, a_n) = \xi[a_1/x_1, a_2/x_2, \ldots, a_n/x_n]$, *where the substitution of objects for variables is simultaneous.*

Notation 4.7 Whenever φ is a wff open in x, we understand $\langle\varphi|x\rangle$ to be the function f in $\mathcal{F}_1$ such that for any a in $\mathcal{F}_0$, $f(a) = \varphi[a/x]$. Since we have full abstraction, we can assume that f exists.

Definition 4.8 *We take* $|\ | : \mathcal{F}_0 \longrightarrow \mathcal{F}_1$ *to be the functional such that*
$| a | = \langle\ \mathbf{app}(a,x)\ |\ x\rangle$.

The following Lemma relates λ to $|\ |$:

Lemma 4.9

-bb-error =

Figure 1: The functional λ

1. $\mid \lambda f \mid = f$

2. $\lambda \mid E \mid = E$.

Proof: *Easy.* $\square$

Of course it is obvious that $\mid\ \mid$ will be used to interpret $\uparrow$. It is the case too that Axiom (1) is satisfied.

Up to now, we can use $\mathcal{F}_0$ to interpret expressions of the following form: $x, c, \lambda f, E'E'', f(E)$ and can use $\mathcal{F}_1$ to interpret all the basic functions and those of the form $\uparrow E$ and $x.E$. Axioms (1) and (2) are satisfied in a Frege structure. Similarly, Axiom (3) is satisfied due to Definition 4.6 and the fact that a Frege structure is a model of the type free λ-calculus.

For the expression ΔE, there exists a functional $\Delta : \mathcal{F}_0 \longrightarrow \mathcal{F}_0$ which obeys the following schema:

For every expression ξ of the Frege structure, $\Delta\xi$ is in **TRUTH** iff ξ is **PROP**.

Of course Δ will be used to interpret Δ, and it will become obvious how axiom (4) is valid in the Frege structure.

The other expressions that we have not interpreted yet are those which involve logic. But this is exactly how a Frege structure departs from a standard model of the type free λ-calculus. It builds logic on the top of a model of the type free λ-calculus. We know that such a construction is not straightforward; for instance, logic cannot be built in a simple way on the top of Scott domains (see [Kamareddine 1992C]). So far, we have said that we have the collections **PROP** and **TRUTH**, but we have not said what they contained and how they were constructed in the first place. The construction provided by Aczel inductively increases the two basic collections of propositions and truths which initially start as being $\{\mathbf{0,1}\}$ and $\{\mathbf{1}\}$ respectively. Aczel takes a set of functionals on the Frege structure which are precisely those logical operators. For example, we have the following

$\forall$: $\mathcal{F}_1 \longrightarrow \mathcal{F}_0$

$\Rightarrow$: $\mathcal{F}_0 \times \mathcal{F}_0 \longrightarrow \mathcal{F}_0$

...

The various logical operators are used to add new elements to the initial collections of propositions and truths, so that for example, $1 \wedge 1$ is added to the collection of truths, $1 \Rightarrow 0$ is added to the set of propositions and so on, obatining the following level of propositions and truths. Again the logical connectives are used to further increase these sets and this process repeats until the fixed point theorem is applied to provide the limit of these newly obtained collections, resulting in **PROP** and **TRUTH**. Hence **PROP** is closed under all the logical connectives $\vee, \forall, \Rightarrow \ldots$ (more strictly, the functionals corresponding to these connectives) and **TRUTH** is the collection of all true propositions. The organization of $\mathcal{F}_0$ and $\mathcal{F}_1$ in a Frege structure is illustrated in Figure 1.

Now that we have some idea of the structures' form, let us try to give an intuitive picture. A Frege structure is a collection of both objects and functions (which are distinct) where we can map any function f into an object a and this object will preserve some of the properties of the function. For instance if the function f is a propositional function, then the nominal of the function, λf, is an object which belongs to the category **SET**.

Definition 4.10 *A* Frege Structure *is a triple* $\mathcal{F} = \langle \mathcal{F}_0, \mathbf{PROP}, \mathbf{SET} \rangle$ *constructed as above.*

4.2 The interpretation of $\mathcal{T}_s$

Definition 4.11 *(Models)*
A model *is a 3-tuple* $\mathcal{M} = \langle \mathcal{F}, \mathcal{I}, g \rangle$*, where*

1. $\mathcal{F}$ *is a Frege Structure ,*
2. $\mathcal{I}$ *is an interpretation function which takes any constant of Exp or basic function to an object or a function in* $\mathcal{F}$*.*
3. g *is an assignment function which takes any variable to* $\mathcal{F}_0$*.*

We now define a valuation function $\llbracket\ \rrbracket$ which given an expression E in Exp or a function f in Fun, and an assignment g yields a value in $\mathcal{F}$.

1. $\llbracket T \rrbracket_{\mathcal{M},g} = \mathbf{1}$
2. $\llbracket \bot \rrbracket_{\mathcal{M},g} = \mathbf{0}$
3. $\llbracket x \rrbracket_{\mathcal{M},g} = g(x)$
4. $\llbracket c \rrbracket_{\mathcal{M},g} = \mathcal{I}(c)$
5. $\llbracket \lambda f \rrbracket_{\mathcal{M},g} = \lambda \llbracket f \rrbracket_{\mathcal{M},g}$
6. $\llbracket E'E'' \rrbracket_{\mathcal{M},g} = \mathbf{app}(\llbracket E' \rrbracket_{\mathcal{M},g}, \llbracket E'' \rrbracket_{\mathcal{M},g})$
7. $\llbracket f(E') \rrbracket_{\mathcal{M},g} = \llbracket f \rrbracket_{\mathcal{M},g}(\llbracket E' \rrbracket_{\mathcal{M},g})$
8. $\llbracket \Delta E \rrbracket_{\mathcal{M},g} = \mathbf{\Delta} \llbracket E \rrbracket_{\mathcal{M},g}$
9. $\llbracket E' \vee E'' \rrbracket_{\mathcal{M},g} = \llbracket E' \rrbracket_{\mathcal{M},g} \vee \llbracket E'' \rrbracket_{\mathcal{M},g}$
10. $\llbracket E' = E'' \rrbracket_{\mathcal{M},g} = (\llbracket E' \rrbracket_{\mathcal{M},g} = \llbracket E'' \rrbracket_{\mathcal{M},g})$
11. $\llbracket E' \Rightarrow E'' \rrbracket_{\mathcal{M},g} = \llbracket E' \rrbracket_{\mathcal{M},g} \Rightarrow \llbracket E'' \rrbracket_{\mathcal{M},g}$
12. $\llbracket \forall x.E' \rrbracket_{\mathcal{M},g} = \forall \mathbf{f}$ where $\mathbf{f} \in \mathcal{F}_1$ and $\mathbf{f}(a) = \llbracket E' \rrbracket_{\mathcal{M},g[a/x]}$
13. $\llbracket f \rrbracket_{\mathcal{M},g} = \mathcal{I}(f)$ if f is basic
14. $\llbracket \uparrow (E) \rrbracket_{\mathcal{M},g} = \mid \llbracket E \rrbracket_{\mathcal{M},g} \mid$
15. $\llbracket x.E \rrbracket_{\mathcal{M},g} = \mathbf{f} \in \mathcal{F}_1$ such that $\mathbf{f}(a) = \llbracket E \rrbracket_{\mathcal{M},g[a/x]}$ for all $a \in \mathcal{F}_0$.

It would be straightforward to convert the clauses for propositions into truth-theoretic definitions, along the following lines:

9′ $\llbracket E' \Rightarrow E'' \rrbracket_{\mathcal{M},g} \in \mathbf{TRUTH} \iff \llbracket E'' \rrbracket_{\mathcal{M},g} \in \mathbf{TRUTH}$ whenever $\llbracket E' \rrbracket_{\mathcal{M},g} \in \mathbf{TRUTH}$

10′ $[\![\forall x.E']\!]_{\mathcal{M},g} \in$**TRUTH** $\iff$ $[\![E']\!]_{\mathcal{M},g[a/x]} \in$**TRUTH** for all a.

Theorem 4.12 *If E is in Exp then $[\![E]\!]_{\mathcal{M},g}$ is in $\mathcal{F}_0$ and if f is in Fun then $[\![f]\!]_{\mathcal{M},g}$ is in $\mathcal{F}_1$.*
Proof: *By straightforward induction.* $\Box$

5 Interpretation

Now let us talk about the interpretation of natural language constructs in this logic. We take the basic plurals to be definable by the function they represent. For example *females* is definable by the propositional function *female*, *students* is definable by the function *student* and so on. Of course from this we can deduce results about combinations of these plurals, for example, *female students* is definable by the function $x.(female(x) \wedge students(x))$.

Singulars are singleton sets which have (apart from their identifying numeral) themselves as member. There are two ways to construct plurals; that is by using $+$ or by using $\cup$. The following examples will illustrate the point

Example 5.1 *John and Mary* where $John = \{John, 1\}$ and Mary = { Mary, 2 } can be represented as John + Mary or as John $\cup$ Mary. By definition, the first is equal to $\lambda x.(x = John \vee x = Mary)$ and hence it is a set which contains John and Mary and only those individuals. The second is equal to $\{John, Mary, 1, 2\}$.

Example 5.2 *John and the boys* which are represented respectively by John = {John,1} and λboy, can be represented as $John + \lambda boy$ which is $\{John, \lambda boy\}$ and as $John \cup \lambda boy$ which is $\{John, 1, a, b, c\}$ if $\lambda boy = \{a, b, c\}$.

Example 5.3 *The boys and the girls* can be represented as $\lambda boy + \lambda girl$ which is $\{\lambda boy, \lambda girl\}$ or as $\lambda boy \cup \lambda girl$ which is the usual union of the sets of boys and girls.

Having set out the background, we are ready to embark on the work of interpreting collectivity and distributivity in a unified way. First, we need to decide for each function whether it is going to be distributive or collective, whether it is going to be singular or mixed.[3] We shall also need the following restriction:

Restriction 5.4 No collective singular predicate can apply to a construct in a distributive way if the construct has the form λf and is not an individual.

Remark 5.5 The distinction between collective/distributive and between singular/mixed is purely semantic. At the start, there is this great semanticist who decides for all verbs whether they are collective/distributive and singular/mixed. Now, for any verb f, $*f$ will be its plural. Hence, nothing is syntactic except $*$. Everything is semantic and set up at the start. This is why I give for each verb, its status. Usually we don't go wrong. For example, *studied*, *play* are fine to assume as distributive and *ate* or *lazy* are fine to assume collective. Furthermore, *ate* is fine to be mixed as the children eat together or alone. Yet, *lazy* really should be singular. So this is really a matter of deciding from the start what one wants.

---

[3] I will not need the case of a purely plural predicate here.

Here is a list of our basic functions and our semantic assumptions:

1. *ate-a-cake* is collective and mixed.
2. *built-a-boat* is mixed.
3. *carry-the-piano* is collective and mixed.
4. *lazy* is collective and singular.
5. *play* is distributive and mixed.
6. *run* is mixed.
7. *studied* is distributive and singular.

Remark 5.6 Some of the verbs above (built-a-boat, run) are not singled out as collective or distributive. The reason for this will become obvious in Section 6.

In conjunction to all above, the following 2 tables will decide on the possible readings of sentences. D and C stand for collective or distributive respectively, S-function and M-function stand for singular respectively mixed function. For a function $f, *f$ stands for the plural predicate of f.

Table 5.7

	1	2	3	4
M-function f	D	D	C	C
reading	D	C	D	C
form	$*f(c)$	$f(c)$	$*f(c)$	$f(c)$

Table 5.8

	1	2	3	4
S-function f	D	D	C	C
reading	D	C	D	C
form	$*f(c)$	$f(c)$	$*f(c)$	–

These tables say that mixed and singular functions act the same except when the function and the reading are both collective. Here is what to do with these tables given a sentence:

1. Look at the main verb of the sentence to see if it is singular or mixed in the above semantic classification. Choose a table according to the status mixed or singular.
2. Now that you know which table you are working in, check again the semantic assumptions to see if the verb is collective or distributive. Pick either columns 1 and 2 or columns 3 and 4 of your table according to the collectivity or distributivity of the verb.
3. Now, you know your table and your columns. Write the one (or two forms) on the bottom line of your two columns. These forms tell you how to write your sentence in the theory (using *f or f). Each of these form is also associated with the unique reading for that form which is given in the middle line of your column.

Example 5.9 *(Table 5.7, columns 1 and 2)*
Let *children* be a collection which contains *John. Anne and Susan* can be represented by one of the following:

- $Anne + Susan = \lambda x.(x = Anne \vee x = Susan)$
- $Anne \cup Susan = \lambda x.(x \in Anne \vee x \in Susan)$

The children play can be interpreted as *Each child plays* (and so we would be able to say that *John plays*) or as *The children together play.* Here is how this is seen from the tables. The children play can now be represented as

1. $*play(\lambda child)$ from Table 5.7, column 1.
2. $play(\lambda child)$ from Table 5.7, column 2.

1 is the distributive reading which is equivalent to *Each child plays* and so if John is a child, we can deduce that: play(John). 2 is the collective reading which says that *The children together play.*

Example 5.10 *(Table 5.7, columns 3 and 4, Lemma 2.13)*
Take the sentence *Anne ate a cake.* We can represent it by either:

1. $*$ate-a-cake$(Anne)$ from Table 5.7, column 3.
2. ate-a-cake$(Anne)$ from Table 5.7, column 4.

Both representations are equivalent according to Lemma 2.13.

Example 5.11 *(Table 5.7, columns 3 and 4)*
Take *The girls ate a cake.* We can represent it by either:

1. $*$ate-a-cake$(\lambda girl)$ from Table 5.7, column 3.
2. ate-a-cake$(\lambda girl)$ from Table 5.7, column 4 .

The above two representations give us the readings we require, that is:

1. Distributive: Each girl ate a cake.
2. Collective: The girls ate a cake between them.

Example 5.12 *(Table 5.8, columns 1 and 2)*
Take *the students studied.* This gives us the forms:

1. $*studied(\lambda student)$ distributive reading, Table 5.8, column 1.
2. $studied(\lambda student)$ collective reading, Table 5.8, column 2.

Example 5.13 *(Table 5.8, columns 3 and 4)*
Dogs are lazy. The only representation is:

$$*lazy(\lambda dog)$$

This is the distributive reading and we would read it as: *Every member of* λ *dog is lazy.*

So to summarize, when we are confronted with a sentence, we look at its main verb to find how our semantic classification treats it. If the verb is mixed we work with Table 5.7, otherwise we work with Table 5.8. Now that we know which table to consider, we look again at the verb to see which columns to consider. If the verb is distributive, we consider columns 1 and 2 otherwise we consider columns 3 and 4. We write the possible forms of the sentence using the bottom line of our columns. The middle line of our columns tells us which readings correpsond to which form.

This is indeed very nice. It tells us what are the possible logical forms of our sentence, and what reading is associated with what form. Furthermore, the various lemmas established about our theory in Section 2, help us to draw some equivalences between these readings (Example 5.10). In the next section, we consider further illustrations.

6 Further Examples

Example 6.1 Take the construct *Jack and the girls.* We have:

- $\lambda girl \cup Jack$
- $\lambda girl + Jack$

Take *Jack and the girls carried the piano.* If we take the representation to be in terms of $\cup$, then we have from Table 5.7, columns 3 and 4, the two possible readings,

1. carry-the-piano$(Jack \cup \lambda girl)$ collective reading.
2. $*$carry-the-piano$(Jack \cup \lambda girl)$ distributive reading.

1 is equivalent to *Jack and the girls carried the piano together*, which is an acceptable form. In the case of 2, the plural predicate **carry-the-piano* distributes over the collection $(Jack \cup \lambda girl)$ giving:

$*$carry-the-piano$(Jack \cup \lambda girl) = *$carry-the-piano$(Jack) \wedge *$carry-the-piano$(\lambda girl)$

$*$carry-the-piano$(\lambda girl)$ is acceptable as it means that each of the girls carried the piano seperately but what about $*$carry-the-piano$(Jack)$? For this we run to Lemma 2.13 which says that $*f(In) \Rightarrow f(In)$. Hence

$$*\text{carry-the-piano}(Jack \cup \lambda girl) \Leftrightarrow \text{carry-the-piano}(Jack) \wedge *\text{carry-the-piano}(\lambda girl)$$

which says that *Jack carried the piano and each of the girls carried the piano.*

Now take, $\lambda girl + Jack$. We can now represent the sentence (again using Table 5.7, columns 3 and 4) by:

3. carry-the-piano$(Jack + \lambda girl)$ the collective reading.
4. $*$carry-the-piano$(Jack + \lambda girl)$ the distributive reading.

3 is equivalent to *Jack and the girls carried the piano together*, which is an acceptable form. In the case of 4, the plural predicate $*$carry-the-piano distributes over the collection $(Jack + \lambda girl)$ giving:

$*$carry-the-piano$(Jack + \lambda girl) =$ carry-the-piano$(Jack) \wedge$ carry-the-piano$(\lambda girl)$

which is equivalent to *Jack carried the piano and the girls together carried the piano.*

Hence the possible readings are:

1. Jack and the girls carried the piano together.

2. Jack carried the piano and each of the girls carried the piano.
3. Jack carried the piano and the girls together carried the piano.

Now, let us look again at Table 5.7. We see that the fact that the function is collective or distributive is irrelevant. In fact, in either case we get $*f(c)$ corresponds to the distributive reading and $f(c)$ corresponds to the collective reading. Hence, Table 5.7 can be simplified to the simple table:

Table 6.2

M-function f	*i*	*ii*
reading	*D*	*C*
form	**f(c)*	*f(c)*

Furthermore, we have the following nice lemma:

Lemma 6.3
Let f be a predicate and let I_n, I_m be singletons. Then $*f(I_n + I_m) = *f(I_n \cup I_m)$.
Proof: $*f(I_n+I_m) =^{Lem\ 2.12} f(I_n) \wedge f(I_m) =^{Lem\ 2.13} *f(I_n) \wedge *f(I_m) =^{Lem\ 2.12} *f(I_n \cup I_m)$.
□

Example 6.4 Take the sentence *John and Mary built a boat.* Table 5.7 gives us the following representations depending on how we represent *John and Mary.*

1. $*$built-a-box($John + Mary$) distributive reading.
2. built-a-box($John + Mary$) collective reading.
3. $*$built-a-box($John \cup Mary$) distributive reading.
4. built-a-box($John \cup Mary$) collective reading.

1 is equivalent to 3 according to Lemma 6.3.

Example 6.5 Take the sentence *Jack and Jill ran away.* Represent *Jack and Jill* as $Jack + Jill$. The sentence can now have the representation:

- $run(Jack + Jill)$.
  This is the collective reading and is equivalent to *Jack and Jill together ran away.*
- $*run(Jack + Jill) = run(Jack) \wedge run(Jill)$.
  This is the distributive reading and is equivalent *Jack ran away and Jill ran away.*

Both of these interpretations are what is required. If we take $Jack \cup Jill$, the sentence can be represented as:

- $run(Jack \cup Jill)$.
  This is the collective reading and is equivalent to *Jack and Jill together ran away.*[4]

---

[4] Note that this is the case because $Jack \cup Jill$ contains Jack and contains Jill.

- $*run(Jack \cup Jill) = run(Jack) \wedge run(Jill)$

Note again that $*run(Jack + Jill) = *run(Jack \cup Jill)$.

Example 6.6 The sentence: *The boys and the girls are lazy*, can be represented as:

1. $*lazy(\lambda boy \cup \lambda girl)$.
2. $*lazy(\lambda boy + \lambda girl)$.

As lazy is a singular predicate, 2 is not possible according to Restriction 5.4 because if we unpack $*lazy(\lambda boy + \lambda girl)$ then it will result in $lazy(\lambda boy) \wedge lazy(\lambda girl)$ which is impossible according to the restriction above, as lazy is a singular predicate.

$*lazy(\lambda boy \cup \lambda girl) = *lazy(\lambda boy) \wedge *lazy(\lambda girl)$

Example 6.7 Take the sentence*John and Mary are lazy.* The possible representations are:

1. $*lazy(John + Mary)$.
2. $*lazy(John \cup Mary)$.

The representations 1 and 2 are equivalent.

7 Conclusions

In this paper, an interpretation of sentences was obtained in the framework of a type-free set theory where individuals were taken to be singletons which contain themselves as members. The idea of a singleton enabled us to reduce *f(E) to f(E) in the case where E was an individual.

Two simple tables from which all the readings of the sentence can be obtained have been provided. These tables when used with the lemmas proved in Section 2 give also the equivalences between various readings of sentences.

We can summarize the paper by saying that we provided a nice type-free theory which represents internally sets and the notions of $+$ and $\cup$ necessary for plurals. We showed some basic results about our sets and singletons (Lemmas 2.9, 2.12 and 2.13). Our internal representation of $+$ and $\cup$ together with these results have been used to reflect on the interpretation of sentences that involve plurals. We concluded that the only hard work that has to be done for this interpretation is to semantically decide at the start, the status of our various verbs. Once this is done, two simple tables would guide us with finding the various interpretation of the sentence.

For instance, in the case that the verb is a mixed function, the table is very simple:

Table 7.1

M-function f	*i*	*ii*
reading	*D*	*C*
form	**f(c)*	*f(c)*

So for example, as play is a mixed function, *the children play* has the two forms from our table:

1. *play(λ child)
2. play(λ child)

From the table, we see that 1 corresponds to the distributive reading and hence, *play(λ child) $\Leftrightarrow (\forall x.x \in \lambda\text{child} \Rightarrow \text{play}(x))$.

The table tells us furthermore that 2 is collective and hence it is *the children play together*.

The nice things about the tables is that they do rule out unwanted readings of sentences. On the other hand, the tables might give us two forms which really correspond to some reading. For example, we may get: $*$built-a-box($John \cup Mary$) and $*$built-a-box($John + Mary$). This is where our theory plays a role in equating these forms that have the same meaning such as the two sentences above.

We presented a type free theory which is certainly superior to ZF in dealing with plurals. For example, singletons cannot be so considered in ZF. Furthermore, our theory follows the line of having expressions and logic. Such line has been shown to be important for the semantics of Natural Language.

We provided two tables which decide the acceptable forms corresponding to the various readings for sentences involving distributive/collective predicates. We said that some of these forms can be established equivalent using our theory. Both the theory and the tables are novel and can be used for further studies in the semantics of Natural Language. We are at the moment investigating intensionality in the present theory.

References

[Aczel 1980] Aczel P, Frege structures and the notions of proposition, truth and set; *The Kleene Synposium*, Edited by Barwise et.al, Studies in Logic 101, North-Holland, New York, 31- 60.

[Kamareddine 1988A] Kamareddine F, *Semantics in a Frege structure*; PhD thesis, University of Edinburgh.

[Kamareddine 1992A] Kamareddine, F., λ-terms, logic, determiners and quantifiers, *Journal of Logic, Language and Information*, Volume 1, No 1, 79-103.

[Kamareddine 1992B] Kamareddine, F., Set Theory and Nominalisation, Part I, *Journal of Logic and Computation*, Volume 2, No 5, 579-604.

[Kamareddine 1992C] Kamareddine, F., Set Theory and Nominalisation, Part II, *Journal of Logic and Computation*, Volume 2, No 6, 687-707.

[Kamareddine 1992D] Kamareddine, F., A system at the cross roads of functional and logic programming, *Science of Computer Programming 19*, 239-279.

[Kamareddine, Klein 1993] Kamareddine, F., and Klein, E., Nominalisation, Predication and Type containment, *Journal of Logic, Language and Information 2*, 171-215.

[Kamareddine 199-] Kamareddine, F., A framework of Logic and Polymorphism, in preparation.

[Landman 1989] Landman F, Groups I & II, *Linguistics and Philosophy 12*, 559-605, and 723-744.

[Link 1983] Link G, The logical analysis of plural and mass terms: A lattice theoretic approach; In *Meaning, Use and Interpretation of Language*, De Gruyter, 302-323.

[Link 1986] Link G, Generalised quantifiers and plurals; CSLI, 86-67.

[Scha 1981] Scha, Distributive, collective and cumulative quantification, in Groenendijk, Janssen, and Stokhoh (eds.), *Formal Methods in the Study of Language*, Mathematical Centre Tracts, Amsterdam.

[Scott 1975] Scott D, Combinators and Classes; In *Lambda Calculus and Computer Science*, edited by Corredo Bőhm, LNCS 37, Springer, 1-26.