Donkey Anaphora: Type-Theoretic Semantics with Both Strong and Weak Sums

Zhaohui Luo
Royal Holloway, University of London
zhaohui.luo@hotmail.co.uk

Abstract

Donkey sentences are among the challenging examples that present a difficult problem in compositional logical semantics and their semantic treatment is one of the early applications of dependent type theory to linguistic semantics, where the strong sum Σ, rather than weak sums (as given by traditional existential quantifiers), is used for existential quantification. However, it is known that this method is inadequate because it fails to deal with counting properly. In this paper, we propose to consider the semantics of donkey sentences in a type theory with both strong and weak sums and show that, with both sum operators, donkey sentences can be given adequate semantic interpretations which, in particular, take care of counting properly.

1 Introduction

Donkey sentences, as first studied by Geach (1962) and exemplified in (1), where an anaphoric expression refers to an existentially quantified entity, are among the challenging examples that present a difficult problem in compositional logical semantics.

(1) Every farmer who owns a donkey beats it.

Their studies (and that of trans-sentential anaphora) have led to the development of dynamic semantics such as DRT (Kamp, 1981) and DPL (Groenendijk and Stokhof, 1991) which, however, require one to consider substantial changes of the underlying logical systems.[1] (For a recent summary of the dynamic approach to donkey anaphora, see Brasoveanu and Dotlacil (2021).)

[1]For example, DPL (Groenendijk and Stokhof, 1991) is a rather non-standard logical system: among other things, it is non-monotonic and the notion of dynamic entailment fails to be reflexive or transitive.

In the mid-80s, as one of the early applications of dependent type theory in logical semantics, researchers such as Mönnich (1985) and Sundholm (1986) have proposed to use Martin-Löf's type theory (Martin-Löf, 1984) to deal with donkey anaphora, where Σ-types are employed to represent existentially quantified formulas. Σ-types $\Sigma x{:}A.P(x)$ are also called *strong sums*, as opposed to the traditional existentially quantified formulas $\exists x{:}A.P(x)$ which are called *weak sums*, because from an object of the strong sum, one can obtain its witness, by means of a projection operation, while this is not possible for the weak sum. It is because of the availability of witness projection that an anaphoric reference can be obtained from an object of a Σ-type, while this is not possible for an existential quantification in the traditional case (and hence the problem in the first place). However, it is known that this approach of using Σ-types to deal with donkey anaphora suffers from a problem of counting (Sundholm, 1989; Tanaka, 2015) and fails to provide us an adequate solution. (See §2 for more details.)

In this paper, we contend that the problem of the above type-theoretical approach has come from a double role played by Σ, as an existential quantifier, on the one hand, and as a structural mechanism to represent collections of objects, on the other. These two roles should be separate and played by different type constructors. But in traditional logics (first-order logic or simple type theory) or in Martin-Löf's type theory, only either $\exists$ or Σ exists, not both, and therefore there is no way to consider such a separation. We show that, in a type theory with both strong and weak sums, donkey sentences can be given adequate semantics in which counting is taken into proper account.

Our proposal is also linked to the research on different readings of donkey sentences and, in particular, the strong and weak readings as studied

Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 45–52
Utrecht, The Netherlands (online), 26–27 July, 2021.

by Chierchia (1990) and others. Also, donkey anaphora are closely related to (and, for some researchers, they are examples of) the so-called E-type anaphora, as first studied by Evans (1977, 1980), which may be interpreted by means of descriptions (see, for example, Nouwen (2021) for a recent discussion). It is not surprising that Σ-types are essentially useful in semantic interpretations of donkey sentences since they have close links to descriptions (Martin-Löf, 1984; Carlström, 2005; Mineshima, 2013) and we shall give some brief discussions about this.

Combining strong and weak sums in type theory is a subtle matter that needs us to tread carefully, for otherwise we may easily slip into problems such as inconsistency. We shall discuss this briefly as well.

This is a short version of a paper we plan to write. In this paper, in particular, we shall focus on telling a complete story of this new treatment of donkey anaphora with both strong and weak sums, but shall be brief about or completely omit some related respects.

2 Strong and Weak Sums in Type Theory

In this section, we explain the concepts of weak sums (for example, traditional existential quantifiers) and strong sums (Σ-types) and, using the notion of the cardinality of a finite type, illustrate the counting problem when using only Σ-types to interpret donkey sentences.

Weak sums (existential quantifiers). Under the Curry-Howard propositions-as-types principle (Curry and Feys, 1958; Howard, 1980), traditional existentially quantified formulas are examples of weak sum types of the form $\exists x.P(x)$. In first-order logic, depending on whether it is intuitionistic or classical, the existential quantifier can be introduced directly or defined by means of the universal quantifier together with negation, respectively. In higher-order logic (or simple type theory) as used in Montague's semantics, where there is an impredicative type **t** of all formulas, it can be either directly introduced or defined by means of the universal quantifier as in (2).

(2) $\exists x.P(x) = \forall X{:}\mathbf{t}.\ (\forall x.(P(x) \Rightarrow X)) \Rightarrow X.$

It is known that, given a proof of $\exists x.P(x)$, although one knows that there is an entity such that P holds, in the logical calculus one cannot find out which entity it is. It is because of this that an anaphoric reference to an existentially quantified entity becomes problematic. For example, in a traditional compositional semantics, the donkey sentence (1) would obtain (3) as its interpretation, which is not a well-formed formula since the variable y in $beat(x, y)$ is out of the scope of the existential quantifier.

(3) (#) $\forall x.\ [farmer(x)\ \&\ \exists y.(donkey(y)\ \&\ own(x, y))] \Rightarrow beat(x, y)$

This illustrates the original problem in interpreting donkey sentences, as mentioned at the beginning of Introduction.

Strong sums (Σ-types). Σ is a dependent type constructor. If A is a type and B is a family of types that depend on objects of type A, then $\Sigma x{:}A.B(x)$ is a type, consisting of pairs (a, b) such that a is of type A and b is of type $B(a)$. Σ-types are associated with the projection operators π_1 and π_2 so that, for (a, b) of type $\Sigma x{:}A.B(x)$, $\pi_1(a, b) = a$ and $\pi_2(a, b) = b$. Formally, Σ-types are governed by the inference rules in Appendix A.

Besides being useful mechanisms to organise structures in various applications, Σ-types may also play other roles. For example, in Martin-Löf's type theory, Σ also plays the role of existential quantifier in its logic[2]. Therefore, for instance, the donkey sentence (1) can be interpreted as (4), in which F_Σ, as defined in (5), is the type intended to represent the collection of donkey-owning farmers, where F and D are the types that interpret farmer and donkey, respectively.[3]

(4) $\forall z : F_\Sigma.\ beat(\pi_1(z), \pi_1(\pi_2(z)))$

(5) $F_\Sigma = \Sigma x{:}F\ \Sigma y{:}D.\ own(x, y)$

Σ-types are strong in the sense that from a proof of $\Sigma x{:}A.P(x)$ one can preform the first projection operation to obtain the witness of this 'existentially' quantified formula and it is because of this, if Σ is used as existential quantifier, one can project out its witness from a proof term of the Σ-type, even

[2] This is concerned with intuitionistic philosophy – a strongly minded intuitionist may believe that the witness of a proven existentially quantified formula can be obtained internally in a logical calculus. We omit further discussions here.

[3] In formal semantics based on modern type theories, CNs such as 'farmer' and 'donkey' are interpreted as types (rather than predicates). This was first proposed by Mönnich (1985) and Sundholm (1986) and further elaborated in (Ranta, 1994; Luo, 2012).

outside its scope (the terms $\pi_1(z)$ and $\pi_1(\pi_2(z))$ in (4) are such examples).

The type F_Σ above contains two occurrences of Σ and they play two different roles: the first acts as a structural mechanism to represent the collection of the farmers who own donkeys and the second as the existential quantifier to say that there exists a donkey owned by the farmer concerned. As we shall see below, using Σ to play this double role is problematic. In particular, F_Σ is in fact representing a collection whose cardinality (the number of its objects) is different from that of the collection of donkey-owning farmers and, therefore, the semantic interpretation (4) of (1) is inadequate (Sundholm, 1989; Tanaka et al., 2015).

Counting and cardinality of finite types. When a type A is finite in the sense that it has finitely many objects, it is possible to define its cardinality $|A|$ as the number of its objects. Formally, a type is finite if, for some n, it is isomorphic to $Fin(n)$, the type with exactly n objects – see Appendix B. For example, the cardinality of a finite Σ-type is the number of pairs in the type.

The problem of counting can be illustrated by considering the sentence in (6),[4] where the quantifier Every in (1) is replaced by Most. Its formal semantics by means of Σ-types in Martin-Löf's type theory is given in (7), which can be seen obtained by replacing $\forall$ by the quantifier $Most_S$, which is defined by Sundholm (1989) (S in $Most_S$ for Sundholm) so that, for a finite type A, $Most_S\ x{:}A.P(x)$ is true if, and only if, more than half of the objects in A satisfy P.

(6) Most farmers who own a donkey beat it.

(7) $Most_S\ z : F_\Sigma.\ beat(\pi_1(z), \pi_1(\pi_2(z)))$

Let us now consider the cardinality of F_Σ, as defined in (5). Because of the second Σ in F_Σ, $|F_\Sigma|$ is not that of the collection of donkey-owning farmers; instead, to calculate $|F_\Sigma|$, we'd have to count every triple (x, y, p) of farmers x, donkeys y and proofs p that x owns y. For example, if there are ten farmers, one of whom owns twenty donkeys and beats all of them, and the other nine own one donkey each and do not beat their donkeys. Then, $|F_\Sigma| \geq 29$ (it is an inequality because, if farmer x owns donkey y, there may be more than one proof that x owns y), but the number of farmers who do not beat their donkeys is 9. Therefore, the above semantics (7) of (6) would be true in such a case, which is obviously incorrect.[5]

[4]Thanks to Justyna Grudziñska for a discussion about this example.

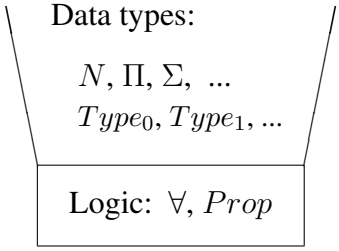

Figure 1: The type structure in UTT.

3 Donkey Anaphora: a Type-Theoretical Solution with Both Σ and $\exists$

In this section, we shall first introduce a dependent type theory UTT (Luo, 1994), which has both strong and weak sums, and then show how donkey sentences like (1) and (6) can be interpreted type-theoretically, giving adequate treatments for different readings and taking care of counting in a proper way as well.

3.1 UTT: an impredicative type theory

The type structure of UTT (Unifying Theory of dependent Types) (Luo, 1994) consists of two parts: the world of data types and that of logical propositions (see Fig. 1). It contains various types such as dependent product types (Π-types), strong sum types (Σ-types), the type N of natural numbers, the universes $Type_i$, and many other types. UTT also contains an impredicative type universe $Prop$ of logical propositions which provide means to describe the logical properties of objects of any type (see Appendix C). Formally, UTT can be considered as the combination of Martin-Löf's (intensional) type theory (Martin-Löf, 1975; Nordström et al., 1990) with Coquand-Huet's Calculus of Constructions (Coquand and Huet, 1988). In computer science, type theories such as UTT have been implemented in theorem proving systems (called proof assistants) for formalisation of mathematics and verification of programs, and recently, they have been used for formal reasoning

[5]This is similar to the 'proportion problem' when one uses DRT to interpret such donkey sentences, where one counts farmer-donkey pairs rather than the donkey-owning farmers. See Kanazawa (1994) and Brasoveanu and Dotlacil (2021), among others, for discussions.

based on linguistic semantics (see, for example, (Chatzikyriakidis and Luo, 2016)).[6]

Note that UTT contains both strong sums $\Sigma x{:}A.B(x)$ (Σ-types, as 'data types') and weak sums $\exists x{:}A.P(x)$ (existentially quantified types, as logical propositions), and this is essential when considering semantic interpretations of donkey sentences in §3.2 below.

Logic and proof irrelevance. In UTT, a type is a logical proposition if it is of type $Prop$. The type universe $Prop$ is impredicative and, therefore, the other logical operators can be defined by means of the operator $\forall$ for universal quantification[7]. For example, the conjunction operator and the existential quantifier $\exists$ can be defined as in (8) and (9), respectively, and the definitions of the other operators can be found in Appendix C.

(8) $P \wedge Q = \forall X : Prop.\ (P \Rightarrow Q \Rightarrow X) \Rightarrow X$

(9) $\exists x : A.P(x)$
$= \forall X{:}Prop.(\forall x : A.(P(x) \Rightarrow X)) \Rightarrow X$

The principle of *proof irrelevance* says that any two proofs of the same logical proposition should be the same. For instance, it implies that, for farmer x and donkey y, any two proof terms of the proposition $own(x, y)$ should be the same. It has been shown that, when employing a type theory for natural language semantics, proof irrelevance should be enforced (Luo, 2012, 2019). Note that, because in UTT there is a clear distinction between logical propositions and other types (the former being those of type $Prop$), it is straightforward to introduce proof irrelevance by means of the following rule (Werner, 2008; Luo, 2012):

$$\frac{P : Prop \quad p : P \quad q : P}{p = q : P}$$

Intuitively, it says that, if P is a logical proposition and if p and q are the same. In particular, according to the above rule, every proposition of type $Prop$ is either an empty type or a singleton type. In terms of cardinality, we have $|P| \leq 1$ for every $P : Prop$ and, therefore, if A is finite and $Q : A \to Prop$ is a predicate over A, then we have

(10) $|\Sigma x{:}A.Q(x)| \leq |A|$.

3.2 Semantics of donkey anaphora in UTT

When a type theory has both strong and weak sums (Σ-types and $\exists$-propositions as in UTT), together with proof irrelevance, there is a new way to semantically interpret donkey sentences, which takes care of counting adequately. We'll use the example (6), which is repeated as (11) below, to explain.

(11) Most farmers who own a donkey beat it.

In §2, we have shown that, because in Martin-Löf's type theory Σ is used to play a double role, the semantic interpretation (7) of (11) is inadequate because it gets counting wrong. In that definition, we have used quantifier $Most_S$ defined in Martin-Löf's type theory and, here, we can define a semantic interpretation of the quantifier most in UTT in a similar fashion as in (Sundholm, 1989) but with a crucial difference: instead of Σ, we shall use $\exists$ as defined in (9) as the existential quantifier and, intuitively, for a finite A, $Most\ x{:}A.P(x)$ also means that more than half of the objects in A satisfy P. Note that, $Most_S\ x{:}A.P(x)$ is a non-propositional type, but $Most\ x{:}A.P(x)$ is a logical proposition of type $Prop$. (See Appendix D for details.)

Having defined $Most$ in UTT, we can now interpret the donkey sentence (11) as (12), in which $F_\exists$ is defined in (13):

(12) $Most\ z : F_\exists.$
$\forall y' : \Sigma y{:}D.own(\pi_1(z), y).\ beat(\pi_1(z), \pi_1(y'))$

(13) $F_\exists = \Sigma x{:}F.\ \exists y{:}D.own(x, y)$

Note that $|\exists y{:}D.own(x, y)| \leq 1$, that is, if $\exists y{:}D.own(x, y)$ is true, the cardinality of the proposition is 1. Therefore, the type $F_\exists$ correctly represents the collection of donkey-owning farmers, as intended, and the above semantics (12) is adequate and, in particular, it deals with counting correctly.

Researchers have studied different readings (in particular, strong and weak readings) of donkey anaphora, as studied by Chierchia (1990, 1992) and

[6] There are several proof assistants based on type theories including Agda (Agda, 2008) based on Martin-Löf's type theory, Coq (Coq, 2010) implementing the type theory pCIC, and Lego/Plastic (Luo and Pollack, 1992; Callaghan and Luo, 2001) implementing UTT. It may be worth remarking that pCIC, implemented in the Coq proof assistant, is very similar to UTT – this is especially the case after Coq's universe Set became predicative in 2004 (it was impredicative in earlier versions).

[7] The fact that other logical operators can be defined in higher-order logical systems by means of universal quantifier was discovered in the 60s by Prawitz (1965) (and several others, independently) and, this is the same in an impredicative type theory.

others. For instance, the strong and weak readings of (11) are (14) and (15), respectively:

(14) Most farmers who own a donkey beat *the donkeys they own*.

(15) Most farmers who own a donkey beat *some donkeys they own*.

The above interpretation (12) of (11) is a strong one, interpreting (14) directly: most donkey-owning farmers beat *all* donkeys they own. A weaker interpretation of its weak reading (15) would be (16), obtained from (12) by changing $\forall$ into $\exists$:

(16) $Most\ z : F_{\exists}.$
$\exists y' : \Sigma y{:}D.own(\pi_1(z), y).\ beat(\pi_1(z), \pi_1(y'))$

People have also considered more sophisticated examples where donkey anaphora are involved in various ways. For example, (17) is one of them, taken from Brasoveanu's thesis (Brasoveanu, 2007), in which the readings for the donkey anaphora are different ('a TV' having a strong reading and 'a credit card' a weak one). Its type-theoretical semantics with both strong and weak sums is given in (18).

(17) Every person who buys a TV and has a credit card uses it to pay for it.

(18) $\forall z : \Sigma x{:}Person.\ \exists y_1{:}TV.\ buy(x, y_1)$
$\qquad\qquad \wedge\ \exists y_2{:}Card.\ own(x, y_2)$
$\forall y : \Sigma y_1{:}TV.\ buy(\pi_1(z), y_1)$
$\exists y' : \Sigma y_2{:}Card.\ own(\pi_1(z), y_2).$
$\quad pay(\pi_1(z), \pi_1(y), \pi_1(y'))$

One may change the quantifier Every in (17) into Most (and make other minor changes in the sentence) and, in that case, we can use the quantifier $Most$ defined in UTT to interpret the sentence and the resulting interpretations take care of counting correctly as well.

3.3 E-type anaphora

Here, we discuss, albeit rather briefly, the so-called E-type anaphora to which donkey anaphora are closed related (and, for some researchers, donkey anaphora are examples of E-type anaphora).[8] E-type anaphora are first studied by Evans (1977, 1980), and further discussed by many, including (Heim and Kratzer, 1998) among others. They can be interpreted by means of descriptions (Russell, 1905, 1919) (see, for example, Nouwen (2021) for a recent discussion). An example, due to Evans, is (19). Note that the pronoun 'they' in (19) is not bound by 'Few' for otherwise the meaning is incorrect. A common conceptual answer, proposed by Evans (1977, 1980), is that these pronouns are *descriptive* in that they can be paraphrased by means of descriptions as exemplified in (20) that paraphrases (19).

(19) Few congressmen admire Kennedy, and they are very junior.

(20) Few congressmen admire Kennedy, and *the congressmen that do admire Kennedy* are very junior.

As pointed out by Martin-Löf (1984), strong sum types (Σ-types) are related to descriptions, because he regards them as logical propositions as well. If you think that $\Sigma x{:}A.B(x)$ as the existentially quantified formula, it is strong and therefore its first projection operator π_1 gives us an internal means of obtaining the witness from a proof of the existentially quantified formula. As explained in §2, this is stronger than the traditional existential operator $\exists$ for which such a projection operator does not exist, and it is exactly because of this that Σ offers a form of description, as pointed out by Martin-Löf (1984) and further studied by Carlström (2005) and Mineshima (2013). For example, the E-type example (19) may be interpreted as (21), either in Martin-Löf's type theory or in UTT, where we assume that the quantifier Few has been defined:

(21) $Few\ x{:}C.\ admire(x, K)$
$\qquad \wedge \forall z{:}[\Sigma x{:}C.admire(x, K)].junior(\pi_1(z))$

However, it should be made clear that Σ-types are not the same as traditional existentially quantified formulas and, therefore, it is unclear how far one may go to analyse E-type anaphora by means of Σ-types. Actually, it would not go very far since, as analysed above, using Σ as existential quantifier does cause problems such as counting, which will show up in context of E-type anaphora as well.

4 Combining Strong and Weak Sums

It is worth mentioning that combining the strong sum (Σ) and the weak sum ($\exists$) in type theory is a subtle matter and, if not careful, it is easy to get into problems. It would be interesting to note that

[8] Here, I use the term 'E-type' for a kind of anaphora, rather than an approach to solving anaphora ('the E-type approach' as people often put it).

UTT does not have 'Σ-propositions' because the so-called 'large Σ-propositions' would lead to inconsistency and the so-called 'small Σ-propositions' would make the weak sum types become strong.[9] Consider, for example, to add large Σ-types into the impredicative universe $Prop$ by adding the following rule (together with those for its introduction and projections that we omit):

$$(*) \qquad \frac{A\ type \quad P : A \to Prop}{\Sigma x{:}A.P(x) : Prop}$$

It turns out that such Σ-propositions cannot be consistently added – if they were added using the above rule $(*)$ (and related ones), the resulting type theory would be inconsistent in the sense that even the false proposition would become provable (Hook and Howe, 1986; Luo, 1994).

One may want to add Σ-propositions (so-called small Σ-types) by a rule like the following, this time restricting A to be a proposition of type $Prop$:

$$\frac{A : Prop \quad P : A \to Prop}{\Sigma x{:}A.P(x) : Prop}$$

Although the resulting type theory may be consistent[10], there is another problem: the addition of such small strong sum as propositions in $Prop$ would make the weak sum proposition $\exists x{:}A.P(x)$ become strong (rather unexpectedly!) in the sense that there is now an internal function in the type theory that, from a proof of $\exists x{:}A.P(x)$, returns an object $a : A$ such that $P(a)$ holds. That would mean that the traditional existential quantifier is not weak anymore – such a side effect is of course problematic and would make the above interpretation method we have proposed fail to deal with counting correctly.

Therefore, neither of the above large or small Σ is a viable possibility and, put in another way, the approach taken in UTT seems to be the only viable approach in combining strong and weak sums.

5 Concluding Remarks

As a concluding remark, we point out that, in this paper, we have studied a completely proof-theoretic approach. This is rather different from the model-theoretic approaches that have been considered in the literature (see, for example, Brasoveanu and Dotlacil (2021)). Among other things, this has the advantage of clearer treatment, on the one hand, and enables the use of proof assistants in reasoning, on the other.

Acknowledgement. I am very grateful to the anonymous reviewers for their helpful remarks.

[9] These situations are discussed in (Luo, 1994), from which the interested reader may obtain more information.

[10] This consistency is a folklore – most researchers, including the author, believe that it is the case, although the author has not seen a proof of it.

References

Agda. 2008. The Agda proof assistant (v2). URL: `http://appserv.cs.chalmers.se/users/ulfn/wiki/agda.php`

A. Brasoveanu. 2007. *Structured Nominal and Modal Reference*. Ph.D. thesis, The State University of New Jersey.

A. Brasoveanu and J. Dotlacil. 2021. Donkey anaphora: farmers and bishops. *In D. Gutzmann et al. (eds), The Wiley Blackwell Companion to Semantics*.

P. Callaghan and Z. Luo. 2001. An implementation of LF with coercive subtyping and universes. *Journal of Automated Reasoning*, 27(1):3–27.

J. Carlström. 2005. Interpreting descriptions in intensional type theory. *The Journal of Symbolic Logic*, 70(2).

S. Chatzikyriakidis and Z. Luo. 2016. Proof assistants for natural language semantics. In *International Conference on Logical Aspects of Computational Linguistics*, pages 85–98. Springer.

G. Chierchia. 1990. Anaphora and dynamic logic. *ITLI Publication Series for Logic, Semantics and Philosophy of Language, LP90-07*.

G. Chierchia. 1992. Anaphora and dynamic binding. *Linguistics and Philosophy*, 15.

Coq. 2010. *The Coq Proof Assistant Reference Manual (Version 8.3), INRIA*.

T. Coquand and G. Huet. 1988. The calculus of constructions. *Information and Computation*, 76(2-3):95–120.

H. Curry and R. Feys. 1958. *Combinatory Logic*, volume 1. North Holland Publishing Company.

G. Evans. 1977. Pronouns, quantifiers and relative clauses. *Canadian Journal of Philosophy*, 7.

G. Evans. 1980. Pronouns. *Linguistic Inquiry*, 11(2).

P. Geach. 1962. *Reference and Generality: An Examination of Some Medieval and Modern Theories*. Cornell University Press.

J. Groenendijk and M. Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy*, pages 39–100.

I. Heim and A. Kratzer. 1998. *Semantics in Generative Grammar*. Blackwell.

J. Hook and D. Howe. 1986. Impredicative strong existential equivalent to Type:Type. Technical Report TR86-760, Cornell University.

W. Howard. 1980. The formulae-as-types notion of construction. In *To H. B. Curry: Essays on Combinatory Logic*, pages 479–490. Academic Press. (Notes written and distributed in 1969.).

H. Kamp. 1981. A theory of truth and semantic representation. *In J. Groenendijk et al (eds.) Formal Methods in the Study of Language*, pages 189–222.

M. Kanazawa. 1994. Weak vs. strong readings of donkey sentences and monotonicity inference in a dynamic setting. *Linguistics and Philosophy*, 17(2).

Z. Luo. 1994. *Computation and Reasoning: A Type Theory for Computer Science*. Oxford University Press.

Z. Luo. 2012. Common nouns as types. In *Logical Aspects of Computational Linguistics (LACL'2012). LNCS 7351*.

Z. Luo. 2019. Proof irrelevance in type-theoretical semantics. *Logic and Algorithms in Computational Linguistics 2018 (LACompLing2018), Studies in Computational Intelligence (SCI)*, pages 1–15. Springer.

Z. Luo and R. Pollack. 1992. LEGO Proof Development System: User's Manual. LFCS Report ECS-LFCS-92-211, Dept of Computer Science, Univ of Edinburgh.

P. Martin-Löf. 1975. An intuitionistic theory of types: predicative part. In *Logic Colloquium'73*.

P. Martin-Löf. 1984. *Intuitionistic Type Theory*. Bibliopolis.

K. Mineshima. 2013. *Aspects of Inference in Natural Language*. Ph.D. thesis, Keio University.

U. Mönnich. 1985. *Untersuchungen zu einer konstruktiven Semantik fur ein Fragment des Englischen*. Habilitation. University of Tübingen.

B. Nordström, K. Petersson, and J. Smith. 1990. *Programming in Martin-Löf's Type Theory: An Introduction*. Oxford University Press.

R. Nouwen. 2021. E-type pronouns: congressmen, sheep and paychecks. *In D. Gutzmann et al. (eds), The Wiley Blackwell Companion to Semantics*.

D. Prawitz. 1965. *Natural Deduction, a Proof-Theoretic Study*. Lmqvist and Wiksell.

A. Ranta. 1994. *Type-Theoretical Grammar*. Oxford University Press.

B. Russell. 1905. On denoting. *Mind*, 14(56).

B. Russell. 1919. *Introduction to Mathematical Philosophy*. George Allen and Unwin.

G. Sundholm. 1986. Proof theory and meaning. In *Handbook of philosophical logic*, pages 471–506. Springer.

G. Sundholm. 1989. Constructive generalized quantifiers. *Synthese*, 79(1):1–12.

R. Tanaka. 2015. Generalized quantifiers in dependent type semantics. Talk given at Ohio State University.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. 2015. Factivity and presupposition in dependent type semantics. In *Proceedings of TyTLeS, ESSLLI2015*.

B. Werner. 2008. On the strength of proof-irrelevant type theories. *Logical Methods in Computer Science*, 4(3).

A Rules for Σ-types

$$\frac{A\ type \quad x{:}A \vdash B\ type}{\Sigma x{:}A.B\ type}$$

$$\frac{a : A \quad b : [a/x]B \quad x{:}A \vdash B\ type}{(a, b) : \Sigma x{:}A.B}$$

$$\frac{p : \Sigma x{:}A.B}{\pi_1(p) : A} \qquad \frac{p : \Sigma x{:}A.B}{\pi_2(p) : [\pi_1(p)/x]B}$$

$$\frac{a : A \quad b : [a/x]B}{\pi_1(a, b) = a : A} \qquad \frac{a : A \quad b : [a/x]B}{\pi_2(a, b) = b : [a/x]B}$$

B Cardinality of Finite Types

We give the formal definition of finite types. It will use the auxiliary type $Fin(n)$ for which we define first.

The type $Fin(n)$, indexed by $n : N$ with N being the type of natural numbers, consists of exactly n objects and can be specified by means of with the following introduction rules (we omit their elimination and computation rules):

$$\frac{n : N}{zero(n) : Fin(n + 1)}$$

$$\frac{n : N \quad i : Fin(n)}{succ(n, i) : Fin(n+1)}$$

The cardinality of a finite type A, notation $|A|$, is defined to be n if, and only if, A is isomorphic to $Fin(n)$, that is, in the type theory concerned, there is a bijective function between A and $Fin(n)$. In particular, $|Fin(n)| = n$, since the identity function over $Fin(n)$ is bijective.

C Logic in UTT

The logic in UTT[11] consists of the impredicative universe $Prop$, specified by the following rules:

$$\frac{}{Prop\ type} \qquad \frac{P : Prop}{P\ type}$$

and the operator $\forall$ for universal quantification, specified by

$$\frac{A\ type \quad x{:}A \vdash P : Prop}{\forall x{:}A.P : Prop}$$

$$\frac{x{:}A \vdash b : P \quad x{:}A \vdash P : Prop}{\lambda x{:}A.b : \forall x{:}A.P}$$

$$\frac{f : \forall x{:}A.P \quad a : A}{f(a) : [a/x]P}$$

$$\frac{x{:}A \vdash b : P \quad a : A}{(\lambda x{:}A.b)(a) = [a/x]b : [a/x]P}$$

In UTT, other logical operators can be defined by means of $\forall$ and here are some definitions (see, for example, §5.1 of (Luo, 1994)):

$$\begin{aligned}
P \Rightarrow Q &= \forall x : P.\, Q \\
\textbf{true} &= \forall X : Prop.\, X \Rightarrow X \\
\textbf{false} &= \forall X : Prop.\, X \\
P \wedge Q &= \forall X : Prop.\, (P \Rightarrow Q \Rightarrow X) \Rightarrow X \\
P \vee Q &= \forall X : Prop. \\
&\qquad (P \Rightarrow X) \Rightarrow (Q \Rightarrow X) \Rightarrow X \\
\neg P &= P \Rightarrow \textbf{false} \\
\exists x : A.P(x) &= \forall X : Prop. \\
&\qquad (\forall x : A.(P(x) \Rightarrow X)) \Rightarrow X \\
(a =_A b) &= \forall P : A \to Prop.\, P(a) \Rightarrow P(b)
\end{aligned}$$

[11] One can find its definition in §9.2.1 of (Luo, 1994), where it is specified in terms of the logical framework LF.

D *Most* in UTT

Let A be a finite type with $|A| = n_A$, $P : A \to Prop$ a predicate over A, and $Fin(n)$ the types with n objects defined in Appendix B. Then, in UTT, the logical proposition $Most\ x{:}A.P(x)$ of type $Prop$ is defined as follows, where $inj(f)$ is a proposition expressing that f is an injective function:

$$\begin{aligned}
&Most\ x{:}A.P(x) \\
&\quad = \quad \exists k : N.\, (k \geq \lfloor n_A/2 \rfloor + 1) \\
&\qquad \wedge \exists f{:}Fin(k) \to A. \\
&\qquad\quad inj(f) \wedge \forall x{:}Fin(k).P(f(x))
\end{aligned}$$