Natural Language Quantification and Dependent Types

OCHANOMIZU UNIVERSITY

Graduate School of Humanities and Sciences

Ribeka Tanaka

March, 2021

Acknowledgements

I want to express my special appreciation and sincere thanks to my advisor Daisuke Bekki. I first learned logic and linguistics in his class. He showed me a lot of interesting issues through clear explanations, which led me to this research area. I would not have written this thesis without his continuous support and encouragement. I would like to express my special thanks to Koji Mineshima, who spent much time with me discussing my work as a collaborator. He always introduced me to helpful literature. I am also grateful to Kenichi Asai, who is my co-advisor. Discussions with him have always brought me back to the basics.

I thank Yusuke Kubota, who gave me a lot of advice and encouragement. He gave me opportunities to present my work at workshops held at Ohio State University in 2015 and 2016, which were valuable experiences for me. I thank Nicholas Asher, who allowed me to visit his research group in 2014. I spent one month in IRIT, Toulouse, discussing dependent types with him and his collaborators. It helped me deepen my understanding.

I have presented different parts of this thesis in a variety of conferences and workshops and received valuable comments. The material in Chapter 4 and Chapter 5 and some of the material in Chapter 3 was presented in the following conferences and workshops: Student Session at 26th European Summer School in Logic, Language and Information (ESS-

LLI2014), the Kyoto Summer School in Logic, Language and Information (KSSLLI) in 2014, the workshop on “Dynamic Semantics: Modern Type Theoretic and Category Theoretic Approaches” held at Ohio State University in 2015, NYU semantics group, the course on "An Introduction to Dependent Type Semantics" at 28th European Summer School in Logic, Language, and Information (ESSLLI2016), “New Landscapes in Theoretical Computational Linguistics” held at Ohio State University in 2016, the 13th International Workshop on Logic and Engineering of Natural Language Semantics (LENLS13).

I thank Chris Barker, Alastair Butler, Lucas Champollion, Robin Cooper, Robert Levine, Zhaohui Luo, Scott Martin, Larry Moss, Anna Szabolcsi, and Shunsuke Yatabe for their helpful comments and suggestions. The preliminary version of the study in Chapter 4 was joint work with Yuki Nakano, who helped me understand the basics of dependent types at the very beginning. Some material in Chapter 3 and Chapter 4 is partially supported by the funding from the Japan Society for the Promotion of Science, Grant-in-Aid for JSPS Research Fellow.

I thank Youyou Cong for her support throughout my research life in the doctoral course. Encouraging each other with her made me feel relaxed. I also thank Tatsuya Hayashi, Hitomi Hirayama, Shiori Ikawa, Shinya Okano and Yu Tomita. We often had a study group together and discussed many topics on logic and linguistics.

I thank Sadao Kurohashi, who offered me a research position at his laboratory. I also thank members of Kurohashi Lab for their support. Because of their understanding, I was able to continue working on this thesis. I thank Yurina Itoh, Kimi Kaneko, Eriko Kinoshita, Kana Manome, Pascual Martínez-Gómez, Ayako Nakamura, Miho Satoh, Maika Utsugi,

Yukiko Yana, Hitomi Yanaka, and Masashi Yoshikawa. I spent most of my time with them in Bekki Lab. I enjoyed discussions with them. I thank Masaatsu Kasukawa, Keiko Kigoshi, Shinji Miura, and Emiko Naito for their support, kindness, and encouragement.

Finally, I wish to express my thanks to my family for their continuous encouragement.

Contents

Chapter 1

Introduction

This thesis provides an analysis of quantifiers in natural language based on a semantic framework called *dependent type semantics* (henceforth, DTS) (Bekki, 2014; Bekki and Mineshima, 2017).

Quantifiers have been studied extensively in both logic and linguistics. It is well-known that the standard first-order logic has two quantifiers, $\forall$ and $\exists$: a universal quantifier $\forall$ is used to express about *everything* and an existential quantifier $\exists$ is used to express about *something*. To account for other quantification, e.g., *most*, *at least 5*, *exactly 3*, and so on, the notion of generalized quantifiers (GQs) was developed (Mostowski, 1957; Lindström, 1966). It generalizes the standard universal and existential quantifiers into relations between sets of objects, which enables to account for various other quantifiers, including binary quantifiers that express the relation between two sets.

This model-theoretic conception of GQs has been introduced to studies of model-theoretic semantics of natural language. Natural language quantifiers such as *every*, *some*, and *most* can be regarded as binary quantifiers from the GQ perspective (Barwise and Cooper, 1981). This account

seems to give the adequate meaning of quantifiers: the relationship they represent for the denotation of given noun phrases and the verb phrases. Also, the inferential properties of those quantifiers, such as *monotonicity*, seems to be naturally derived from those set relations.

However, it is not that straightforward because natural language quantifiers behave in a complicated way, interacting with context-dependent dynamic linguistic phenomena such as anaphora and presupposition. For example, anaphora in a sentence poses a challenge to the basic idea that a quantifier represents a relationship between two sets. The following example shows the so-called donkey sentence (Geach, 1962) that contains the quantifiers *most*[1].

(1.1) Most farmers who own a^1 donkey beat it_1.

Here, a semantic dependency exists between the two sets related by the quantifier *most* : the second set, which is a collection of objects x that satisfy x *beat it*, is interpreted dependently on the first set. Therefore, to give proper meaning to a quantifier, it is necessary to consider anaphoric dependency simultaneously. This observation suggests that even the essential operation of quantification, i.e., counting, becomes non-trivial in the presence of intra-sentential anaphora. Also, quantifiers exhibit inferential properties such as monotonicity according to the relationship between the two sets they relate to. This inferential property also manifests itself as an entailment relation between natural language sentences containing quantifiers. Therefore, inferential properties of quantifiers must also be treated simultaneously with intra-sentential anaphora. Besides, we need to consider also quantifiers' anaphoric potential for subsequent

[1]In examples in this thesis, an anaphor is subscripted by an index, while its antecedent is superscripted by the same index.

sentences.

It has been one of the significant issues in natural language semantic studies to capture the meaning of quantifiers that accounts for natural language dynamism. For example, dynamic semantic frameworks such as Dynamic Predicate Logic (DPL) (Groenendijk and Stokhof, 1991; Kanazawa, 1994) and Discourse Representation Theory (DRT) (Kamp and Reyle, 1993; Kamp et al., 2011) have provided their account on quantification and intra-sentential anaphora. Analyses have also been proposed for quantifiers' anaphoric potential in discourse (Krifka, 1996; van den Berg, 1996b).

DTS, the semantic framework we study in this thesis, is a proof-theoretic natural language semantics based on dependent type theory (Martin-Löf, 1984). It has been recently developed as an alternative to model-theoretic dynamic semantics such as DRT and DPL.

DTS has shown unique characteristics as a semantic framework for natural language in at least two points. The first point is how it provides a solution to the *compositionality problem* of discourse anaphora. The problem occurs in the donkey sentences and E-type anaphora case (Evans, 1980), which are exemplified below.

(1.2) Every farmer who owns a^1 donkey beats it_1.

(1.3) A^1 man entered. He_1 whistled.

It has been known to be a puzzle to provide their semantic representations parallel to their syntactic structure. However, as Sundholm (1986) pointed out, the type structures of dependent type theory are rich enough to allow such semantic representations. DTS elaborates this idea further and provides a fully compositional account of anaphora resolution and presupposition binding process.

The second point is that DTS accounts for anaphora resolution and presupposition binding involving *inference*. For the analysis of presupposition, DTS follows the "presupposition as anaphora" paradigm proposed by van der Sandt and Geurts (1991) and van der Sandt (1992) and gives a uniform account of anaphora resolution and presupposition binding. Although van der Sandt's analysis on DRT framework is considered to have best empirical coverage (see Beaver, 1997, page. 983), using world knowledge in presupposition binding is beyond the scope of the proposal (Krahmer and Piwek, 1999). The typical instance of such cases is the so-called *bridging inference* (Clark, 1975).

(1.4) John[1] bought a car. He$_1$ checked the motor.

Here, the definite description *the motor* is associated with *a car* in the first sentence by a general world knowledge that *every car has a motor*. In DTS, anaphora resolution and presupposition binding involve *proof search* under the given context along the paradigm of "anaphora resolution as proof construction" (Krahmer and Piwek, 1999). Thus, it can naturally account for presupposition binding involving inference using world knowledge.

Although DTS looks promising as a new framework of natural language semantics, an analysis of quantifiers other than universal and existential ones within the framework has not been well established so far. There is a study of quantifiers based on dependent type theory by Sundholm (1989). The study provides a set-theoretic definition of quantifiers in the framework of dependent type theory, which provides a basis for analyzing quantifiers' meaning in a constructive setting. However, the interaction of quantifiers with dynamic phenomena in natural language has not been fully discussed.

Since DTS is a proof-theoretic semantics, it is not trivial how to ac-

count for the dynamic meaning of quantifiers in the framework even if there are studies in model-theoretic dynamic semantics. Quantifiers have been studied in the proof-theoretic approach to natural language semantics in general. Those studies, however, have mainly focused on the proof-theoretic account of inferential properties of quantifiers, where anaphora is not taken into account. Based on the previous work by Sundholm (1989), we adopt an approach where counting operations are defined explicitly in the dependent type theory framework. We then show that the semantic representation of quantifiers given in this way can provide a context for anaphora and presupposition in the proof-theoretic setting of DTS.

The structure of this thesis is as follows.

In Chapter 2, we introduce DTS, the semantic framework we adopt in this study. We first explain dependent type theory, which is the basis of the framework. There are two dependent types, Σ-type and Π-type, that correspond to the existential and universal quantifiers. We will explain that these types give semantic representations of existential and universal sentences in natural language. We will then introduce the analysis of anaphora resolution in DTS. We categorize natural language anaphora into two groups. One is anaphora that Σ-types contribute to (Σ-*type anaphora*). The other is anaphora that Π-types contribute to (Π-*type anaphora*). We will explain the basics of the anaphora resolution process in DTS for each of them. Moreover, we will explain the same mechanism applies to the analysis of presupposition.

Chapter 3 gives an overview of studies of natural language quantifiers, particularly phenomena associated with determiners. It summarizes the issues discussed in the literature, including intra-sentential and inter-sentential anaphora and inferential properties associated with the quantifiers.

In Chapter 4, we propose an analysis of quantifiers in DTS. We first introduce a way to handle cardinal numbers in dependent type theory, following the study by Sundholm (1989). Among various quantifiers, we pay special attention to *most* and take it as our working example. From the GQ theory point of view, meanings of binary quantifiers such as *most* are given as relations between two sets. We will show how such a view is formalized in our setting while maintaining its internally-dynamic nature. We generalize our definition also to other quantifiers, including numerals and *some*, *no*, and *every*. We also discuss our account of the so-called existential reading (weak reading) and universal reading (strong reading) of donkey sentences. Finally, we discuss the anaphoric potential of the semantic representation given here. We give a semantic representation of the plural pronoun and examine the interaction between the quantifier, the plural pronoun, and the singular pronoun. We will show that the meaning of quantifiers is directly supplied as the context in discourse.

Chapter 5 is concerned with the inferential property of quantifiers. From a proof-theoretic and dynamic perspective, we define inferential properties such as *conservativity* and *monotonicity* in terms of dependent type theory. We then prove the theorems that the semantic representations of quantifiers given in this thesis satisfy conservativity and right monotonicity. We also discuss the remaining issues on left monotonicity. As the theorems consider anaphoric dependency in a sentence, they can account for natural language inference involving anaphora.

Finally, Chapter 6 summarizes the contribution of this study and some remaining issues.

Chapter 2

Dependent Type Semantics

In this chapter, we introduce the framework of *Dependent Type Semantics* (henceforth, DTS) (Bekki, 2014; Bekki and Mineshima, 2017), a semantic framework we will adopt in this study.

Before introducing a framework of DTS, Section 2.1 provides a brief overview of *dependent type theory* (Martin-Löf, 1984; Nordström et al., 1990), a theoretical background of the framework. In Section 2.2, we explain how the theory has been applied to natural language semantics.

We introduce DTS in Section 2.3 and provide an overview of the framework. The following sections, Section 2.4 to Section 2.8, are devoted to explain the natural language analysis in DTS, including the analysis of anaphora and presupposition.

2.1 Dependent type theory

2.1.1 Types depending on terms

Dependent type theory (Martin-Löf, 1984; Nordström et al., 1990) is originally developed as a formal systems for constructive mathematics (see, e.g., Martin-Löf, 1982; Aczel, 1978, 1982, 1986). It extends simple type theory with the notion of *types depending on terms*, that is, types indexed by objects in another type.[1] For instance, one can define a type **list**(n) of lists of length n, where n is an object of type **Nat** (natural number).

Dependent types enable more precise definitions of types. Suppose we want to define a function that takes a natural number i and returns an $i \times i$ identity matrix. Without dependent types, the type of the function is specified as **Nat** $\rightarrow$ **Mat** with the type **Mat** of matrices. However, this type is not so precise in that it may include functions that return matrices of any size. With dependent types, we can define the type **Mat**(n, m) of $n \times m$ matrices for n : **Nat** and m : **Nat**. Therefore, the function can have type (i : **Nat**) $\rightarrow$ **Mat**(i, i), which specifies that the resultant object is $i \times i$ matrix for the given i : **Nat**.

One advantage of dependent types is that it allows specifying a property of an object in its type. Let us consider the operation of matrices multiplication as an example. Without dependent types, the operation has a function type of **Mat** $\rightarrow$ **Mat** $\rightarrow$ **Mat**. However, the matrices multiplication is not always available: the operation is not executable when

[1] The formalism presented in this thesis is based on Martin-Löf type theory (Martin-Löf, 1984). In this thesis, we use the term *dependent type theory* to refer to any type theory with dependent types, including λP (Barendregt, 1992), Calculus of Construction (CoC) (Coquand and Huet, 1988), and Unified Type Theory (UTT) (Luo, 2012b).

the number of columns of the first argument differs from the second argument's number of rows. With dependent types, we can block such a multiplication beforehand. The type of matrices multiplication is given as follows.

$$(i : \mathbf{Nat}) \to (j : \mathbf{Nat}) \to (k : \mathbf{Nat}) \to \mathbf{Mat}(i, j) \to \mathbf{Mat}(j, k) \to \mathbf{Mat}(i, k)$$

Here, the type itself specifies that the number of columns of the first argument and the number of rows of the second argument are the same (variable j). Therefore, the multiplication that does not satisfy the size requirement can be detected in the type checking stage.

2.1.2 Formal system

There are two type constructors, Σ and Π, in dependent type theory.

The type constructor Σ is a generalized form of the product type. An object of type $(\Sigma x : A)\, B(x)$ is a pair (m, n) such that m is of type A and n is of type $B(m)$. Σ-types are associated with projection functions π_1 and π_2 that are computed with the rules $\pi_1(m, n) = m$ and $\pi_2(m, n) = n$, respectively. Conjunction $A \wedge B$ is a degenerate form of $(\Sigma x : A)\, B$, where x does not occur free in B.

The type constructor Π is a generalized form of the function type. An object of type $(\Pi x : A)\, B(x)$ is a function f such that for any object a of type A, $f\,a$ is an object of type $B(a)$. Implication $A \to B$ is a degenerate form of $(\Pi x : A)\, B$, where x does not occur free in B.

Throughout this thesis, we will make use of the DTS-notation for Π-types and Σ-types as shown in Figure 2.1.[2] The inference rules for Π-types

[2] We use the two different notations of Σ-types depending on the situation.

	Π-types	Σ-types
Standard notation	$(\Pi x : A)\, B$	$(\Sigma x : A)\, B$
Notation in DTS	$(x : A) \to B$	$\begin{bmatrix} x : A \\ B(x) \end{bmatrix}$ or $(x : A) \times B$
When $x \notin fv(B)$	$A \to B$	$\begin{bmatrix} A \\ B \end{bmatrix}$ or $(A \times B)$

Figure 2.1: Notation for Π-types and Σ-types in DTS. Here, $fv(B)$ means the set of free variables in B.

and Σ-types are shown in Figure 2.2. Here we write $B[M/x]$ for the substitution of a term M for free occurrences of the variable x in the term B, with the possible capture-avoiding renaming of bound variables. The full version of the inference rules we will use in this thesis is shown in Appendix A. For more details, the readers can refer to Martin-Löf (1984), Nordström et al. (1990), and Ranta (1994).[3]

The form "$a : A$" that expresses "a is a term of type A" is called a *judgement*. Under the so-called *Curry-Howard correspondence* (the *propositions-as-types* principle), a type and a term can be regarded as a proposition and a proof, respectively. Thus, the judgement $a : A$ can also be read as "a is a proof of proposition A". A term is called a *proof term* in this respect, as it serves as proof of a proposition. In this setting, the truth of a proposition A is defined as the existence of a proof term of type A.

[3] Here is a note on some differences between Martin-Löf (1984)'s original system and systems that the specific version of DTS we adopt here is based on. Firstly, DTS employs two sorts: **type** and kind. Secondly, we use an intensional version of the equality (Nordström et al., 1990). Thirdly, we have β-conversion rules in place of judgemental equality (Bekki and Satoh, 2015).

Formation rules:

For any $(\mathsf{s}_1, \mathsf{s}_2) \in \{(\textbf{type}, \textbf{type}),\ (\textbf{type}, \mathsf{kind}),\ (\mathsf{kind}, \textbf{type}),\ (\mathsf{kind}, \mathsf{kind})\}$, and $(\mathsf{s}'_1, \mathsf{s}'_2) \in \{(\textbf{type}, \textbf{type}),\ (\textbf{type}, \mathsf{kind}),\ (\mathsf{kind}, \mathsf{kind})\}$,

$$\frac{\begin{array}{c}\overline{x : A}\,\mathsf{j} \\ \vdots \\ A : \mathsf{s}_1 \quad B : \mathsf{s}_2\end{array}}{(x : A) \to B : \mathsf{s}_2}\ \Pi F, \mathsf{j} \qquad \frac{\begin{array}{c}\overline{x : A}\,\mathsf{j} \\ \vdots \\ A : \mathsf{s}'_1 \quad B : \mathsf{s}'_2\end{array}}{(x : A) \times B : \mathsf{s}'_2}\ \Sigma F, \mathsf{j}$$

Introduction rules: For any $\mathsf{s} \in \{\textbf{type}, \mathsf{kind}\}$,

$$\frac{\begin{array}{c}\overline{x : A}\,\mathsf{j} \\ \vdots \\ A : \mathsf{s} \quad M : B\end{array}}{\lambda x.M : (x : A) \to B}\ \Pi I, \mathsf{j} \qquad \frac{M : A \quad N : B[M/x]}{\langle M, N\rangle : (x : A) \times B}\ \Sigma I$$

Elimination rules

$$\frac{M : (x : A) \to B \quad N : A}{MN : B[N/x]}\ \Pi E$$

$$\frac{M : (x : A) \times B}{\pi_1(M) : A}\ \Sigma E \qquad \frac{M : (x : A) \times B}{\pi_2(M) : B[\pi_1(M)/x]}\ \Sigma E$$

Figure 2.2: Inference rules: formation rules ($\Pi F, \Sigma F$), introduction rules ($\Pi I, \Sigma I$), and elimination rules ($\Pi E, \Sigma E$).

We use the following form to state that the judgement $a : A$ is derivable under the sequence of assumptions Γ.

$$\Gamma \vdash a : A$$

We use the following judgement of the form $A\ true$ to mean that the type A is *inhabited*, that is, there exists a term a that satisfies $\Gamma \vdash a : A$.

$$\Gamma \vdash A\ \mathit{true}$$

By the Curry-Howard correspondence, the type constructor Σ behaves as an existential quantifier and Π as a universal quantifier. Thus a proof term of an existential proposition is a pair, and a proof term of a universal proposition is a function. These types and proof terms play an important role in representing natural language sentences in dependent type theory.

2.2 Dependent types and natural language

2.2.1 Donkey sentence

Dependent type theory has been applied to natural language semantics, particularly to dynamic semantics and lexical semantics (Sundholm, 1986; Ranta, 1994; Luo, 2012b; Cooper, 2012; Bekki, 2014; Bekki and Mineshima, 2017; Chatzikyriakidis and Luo, 2017; Grudzińska and Zawadowski, 2017). Sundholm (1986) first pointed out that the rich type structure can naturally handle dependencies lying in natural language, especially the dependencies between an anaphoric expression and its reference.

Capturing anaphoric dependencies has been one of the biggest issues in studies of dynamic semantic frameworks of natural language. Let us consider the following sentence, known as a *donkey sentence* (Geach, 1962).

(2.1) Every farmer who owns a^1 donkey beats it_1.

If we focus on one natural reading where the pronoun *it* refers back to an indefinite *a donkey*, the sentence means that for every farmer who owns a donkey, the farmer beats his donkey.

This sentence is known to be problematic in the sense that a standard compositional analysis does not give the truth-conditionally correct se-

mantic representation (the *compositionality problem*)[4]. One might think that the sentence can be represented as the following predicate logic formula.

(2.2) $\forall x(\textbf{farmer}(x) \wedge \exists y(\textbf{donkey}(y) \wedge \textbf{own}(x,y)) \rightarrow \textbf{beat}(x,y))$

The structure of the formula looks well-corresponding to the original natural language sentence. However, this does not capture the intended interpretation of the donkey sentence because the variable y in the object position of the predicate **beat** is outside the scope of existential quantification over the variable y. The problem can be stated in a more general way. In the original sentence, since it states there is an object (i.e., a donkey), the object can be referred to from the rest of the sentence. However, there is no way to refer to the object in the predicate logic formula once the existential quantifier's scope is closed.[5]

This gap between natural language sentences and predicate logic formulas became the motivation behinds the dynamic predicate logic (DPL) (Groenendijk and Stokhof, 1991). In DPL, the meaning of a sentence corresponds to an action performed on a *context*. Since existential quantification is treated as an operation to alter the context, the scope of existential quantifier does not affect the accessibility. In this way, DPL provides a compositional analysis of anaphora in (2.1).

[4] Bekki and Mineshima (2017) discussed the compositionality problem of discourse anaphora, including the donkey sentence and E-type anaphora.

[5] Note that providing the truth-conditionally correct representation of the donkey sentence in the predicate logic itself is not a problem. We can represent its meaning as follows: $\neg\exists x(\textbf{farmer}(x) \wedge \exists y(\textbf{donkey}(y) \wedge \textbf{own}(x,y) \wedge \neg\textbf{beat}(x,y)))$.

Note also that the puzzle presented here comes from the assumptions that (1) the interpretation of the sentences is given one-by-one from left to right and (2) the pronoun is a variable-like semantic object.

However, dependent type theory can provide an alternative solution: it provides a way to access the donkey from outside the existential scope. As Sundholm (1986) presents, the corresponding formula in dependent type theory is as follows.

$$(2.3)\quad \left(u : \left[\begin{array}{l} x : \textbf{farmer} \\ \left[\begin{array}{l} y : \textbf{donkey} \\ \textbf{own}(x, y) \end{array} \right] \end{array} \right] \right) \rightarrow \textbf{beat}(\pi_1 u, \pi_1 \pi_2 u)$$

The formula (2.3) as a whole is a universal sentence, which is represented as a Π-type. The restrictor *farmer who owns a donkey* is analyzed as a Σ-type. Since the variable u has the Σ-type $(x : \textbf{farmer}) \times (y : \textbf{donkey}) \times \textbf{own}(x, y)$, the variable would be a tuple $(f, (d, o))$, where f is a term of type **farmer**, d is a term of type **donkey**, and o is a proof-term of the proposition $\textbf{own}(f, d)$. Recall that π_1 and π_2 are projection functions that take a pair and return the first and the second element, respectively. Thus the terms $\pi_1 u$ and $\pi_1 \pi_2 u$ appearing in the argument position of **beat** pick up from u the term f of type **farmer** and the term d of type **donkey**, respectively. Here, the situation is the same as before in the sense that **beat** is outside the scope of y. However, via the proof term u associated with the Σ-type, the subsequent discourse can access the antecedent.

An advantage of dependent type theory over the previous dynamic theories is that such an externally dynamic character of quantification can be captured without any stipulation: Σ-types and Π-types, which are natural generalizations of existential and universal quantifiers in predicate logic, are equipped with the mechanism to handle dynamic aspects of discourse interpretations.

2.2.2 Toward compositional analysis

Since Sundholm (1986) discovered that dependent type theory could provide an alternative treatment to donkey anaphora, studies of natural language semantics have been developed based on the approach.

One of the notable works was done by Ranta (1994), who developed Type Theoretical Grammar (TTG) and showed the potential of dependent types with broad coverage of empirical data. The study also raised fundamental discussions about a proof-theoretic view of anaphora, including the relation of proof-terms with the notion of reference in dynamic semantics. It shows that dependent type theory can provide a type-theoretic alternative to model-theoretic frameworks such as Dynamic Semantics (Heim, 1983), Dynamic Predicate Logic (Groenendijk and Stokhof, 1991), and Discourse Representation Theory (van der Sandt, 1992; Kamp and Reyle, 1993; Kamp et al., 2011).

While Ranta's work mainly focuses on mapping from a given formula to the corresponding natural language sentence, one of the essential questions to be answered is how to get the semantic representation (2.3) from sentence (2.1) in a systematic way.

From the viewpoint of the standard compositional semantics, the problem can be divided into two tasks. The first is to map the sentence (2.1) into an *underspecified* representation, namely, a semantic representation that contains an underspecified element corresponding to the pronoun in question. The second task is to resolve the underspecified element. In our example (2.3), the underspecified element has been resolved to $\pi_1\pi_2 u$. The semantics satisfying the requirement of compositionality must provide an explicit procedure to do these two tasks.

DTS (Bekki, 2014; Bekki and Mineshima, 2017) provides such a proce-

dure. We will explain these in more detail from the following sections.

2.3 DTS: an overview

DTS (Dependent Type Semantics) (Bekki, 2014; Bekki and Mineshima, 2017) is a proof-theoretic natural language semantics based on dependent type theory (Martin-Löf, 1984), which has been developed as an alternative framework to traditional model-theoretic dynamic semantics. One of the distinctive features of DTS, compared with other frameworks based on dependent type theory, is that it is augmented with underspecified terms called @-terms. DTS utilizes @-terms to provide a unified analysis of entailment, anaphora, and presupposition from an inferential and computational perspective. DTS is designed as a semantic component of modern categorial grammars to give a fully compositional account of inferences involving anaphora.

Figure 2.3 shows the flow of language analysis in DTS. First, each sentence in discourse is translated into its semantic representation by semantic composition. The semantic representation in DTS involves an underspecified representation called an *@-term*. The underlying system is what we call *underspecified dependent type theory (UDTT)*, dependent type theory augmented with an underspecified term. We obtain the semantic representation of the whole discourse by combining sentence representations with progressive conjunction. The process of resolving underspecification is formulated as *type checking*, *proof search*, and *@-elimination*. We finally obtain a fully-specified semantic representation in dependent type theory to conduct standard inference in dependent type theory.

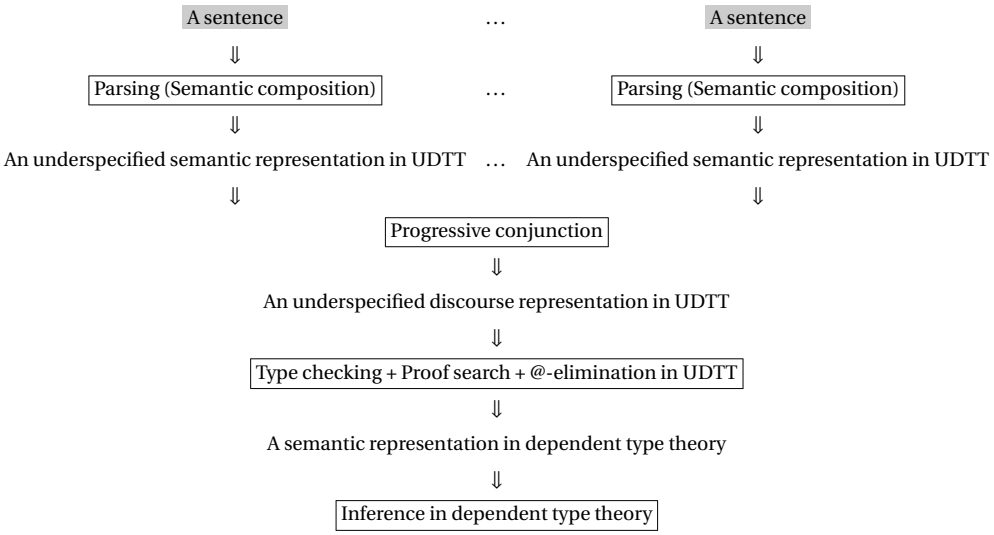


Figure 2.3: The model of language analysis in DTS.

2.4 Semantic representation in DTS

We first explain how semantic representations of basic sentences are derived in the framework. Consider the following two sentences.

(2.4) A man entered.

(2.5) Every man entered.

DTS provides a compositional mapping from natural language sentences to their semantic representations. In order to give an explicit compositional mapping, we adopt Combinatory Categorial Grammar (CCG, Steedman 2000) as a syntactic framework.[6] Table 2.1 shows some of the lexical entries.[7] Here we use **entity** : **type** for the type of entities.

[6] Note that, as is emphasized in Bekki (2014), DTS can be combined with other categorial grammars; see Kubota and Levine (2015) for a concrete proposal that combines DTS with a type-logical grammar.

[7] The subscript *nom* is for determiners in the nominative position. For the purpose

Expression	CCG category	Semantic Representation
a_{nom}	$(S/(S\backslash NP))/N$	$\lambda n\lambda v.\,(x : \mathbf{entity}) \times (u : nx) \times vx$
$every_{nom}$	$(S/(S\backslash NP))/N$	$\lambda n\lambda v.\,(u : (x : \mathbf{entity}) \times nx) \to v(\pi_1 u)$
man	N	$\lambda x.\mathbf{man}(x)$
entered	$S\backslash NP$	$\lambda x.\mathbf{enter}(x)$
own	$(S\backslash NP)/NP$	$\lambda y\lambda x.\mathbf{own}(x,y)$
who	$(N\backslash N)/(S\backslash NP)$	$\lambda v\lambda n\lambda x.\,(nx \times vx)$
he, it	NP	$@_i :: \mathbf{entity}$

Table 2.1: Some lexical entries.

The semantic representations of (2.4) and (2.5) are derived as in (2.6) and (2.7), respectively.[8,9]

---

of concreteness, we use a type-raised form of semantic representations for determiners. The entry for determiners in the object position can be given as follows:

$$a_{acc};\ ((S\backslash NP)\backslash((S\backslash NP)/NP))/N;\ \lambda n\lambda v\lambda x.\,(y : \mathbf{e}) \times (u : ny) \times vyx$$

Although there are other possible syntactic analyses of determiners in object position (cf. Bekki, 2014), this entry would ensure a concise derivation tree for semantic composition.

[8]We ignore tense in this thesis for simplicity.

[9]In the CCG derivation trees, we use two standard combinatory rules, forward (>) and backward (<) function application rules:

$$\frac{X/Y : m \quad Y : n}{X : mn}\ > \qquad \frac{Y : n \quad X\backslash Y : m}{X : mn}\ <$$

For instance, the combinatory rule (>) means that an expression having a syntactic category X/Y and a meaning m, combined with an expression having a syntactic category Y and a meaning n, yields an expression having a category X and a meaning mn. Each meaning is represented as a lambda term. See (Steedman, 2000) for more details.

(2.6)

$$\dfrac{\dfrac{\dfrac{\text{A}}{(S/(S\backslash NP))/N}}{\lambda n\lambda v.\left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : nx \end{array}\right] \\ vx \end{array}\right]} \quad \dfrac{\dfrac{\text{man}}{N}}{\lambda x.\textbf{man}(x)}}{\dfrac{S/(S\backslash NP)}{\lambda v.\left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : \textbf{man}(x) \end{array}\right] \\ vx \end{array}\right]} >} \quad \dfrac{\dfrac{\text{entered}}{S\backslash NP}}{\lambda x.\textbf{enter}(x)}}{\dfrac{S}{\left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right]} >}$$

(2.7)

$$\dfrac{\dfrac{\dfrac{\text{Every}}{(S/(S\backslash NP))/N}}{\lambda n\lambda v.\left(u : \left[\begin{array}{l} x : \textbf{entity} \\ n(x) \end{array}\right]\right) \rightarrow v(\pi_1 u)} \quad \dfrac{\dfrac{\text{man}}{N}}{\lambda x.\textbf{man}(x)}}{\dfrac{S/(S\backslash NP)}{\lambda v.\left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{man}(x) \end{array}\right]\right) \rightarrow v(\pi_1 u)} >} \quad \dfrac{\dfrac{\text{entered}}{S\backslash NP}}{\lambda x.\textbf{enter}(x)}}{\dfrac{S}{\left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{man}(x) \end{array}\right]\right) \rightarrow \textbf{entered}(\pi_1 u)} >}$$

Here is a note on the treatment of common nouns. There are two possible approaches to representing basic sentences like *A man entered* in

dependent type theory. One is the approach proposed in Ranta (1994) and Luo (2012a,b), according to which common nouns like *man* are interpreted as *types* so that the sentence is represented as (2.8) in our notation.

(2.8) $\begin{bmatrix} x : \mathbf{man} \\ \mathbf{enter}(x) \end{bmatrix}$

As an alternative approach, a common noun is interpreted as a predicate in DTS. Common nouns in the argument position and the predicate position are both analyzed as predicates of type **entity** → **type**.

(2.9) A man enters. $\begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix}$

(2.10) John is a man. $\mathbf{man}(\mathsf{john})$

As this approach is in line with the traditional analysis of common nouns, we can integrate standard assumptions in formal semantics into our framework. See Bekki and Mineshima (2017); Chatzikyriakidis and Luo (2017) for more discussions on the treatment of common nouns. [10]

[10] Retoré (2013) suggests the possibility that common nouns can be interpreted both as types and as predicates; for instance, using the type **entity**, the common noun *animal* interpreted as a type **animal** could be related to a predicate $\mathbf{animal}^*$ of type **entity** → **type**, via some suitably defined mapping $(\cdot)^*$ from one to another. The question of whether a type system for natural language semantics needs to be enriched with the structures of common nouns would ultimately depend on the treatment of lexical-semantic phenomena it attempts to capture, such as coercion and selectional restriction – phenomena that have been widely discussed in the recent literature on type-theoretical semantics (Asher and Pogodalla, 2011; Asher and Luo, 2012; Bekki and Asher, 2013; Retoré, 2013; Kinoshita et al., 2016, 2018). Investigating this matter further is beyond the scope of this thesis.

2.5 Analysis of anaphora

In this section, we explain the anaphora resolution process in DTS.

As is well known in the literature, existential sentences and universal sentences have different anaphoric potentials. From the dependent-type perspective, we classify the class of anaphoric phenomena into two groups: Σ-type anaphora and Π-type anaphora.

2.5.1 Σ-type anaphora

An existential quantifier is said to be *externally dynamic* (Groenendijk and Stokhof, 1991) in the sense that the entity introduced by an existential quantifier is accessible to the subsequent sentences. This property can be captured by representing existential quantifiers using Σ-types. We call the type of anaphora in which an object introduced by an existential quantifier is referred to from the subsequent discourse *Σ-type anaphora*.

As an illustration, consider the case of singular E-type anaphora (2.11), a typical case of Σ-type anaphora.

(2.11) a. A^1 man entered.

b. He_1 whistled.

We will focus on the reading where *he* refers back to *a man*, which we indicate by a subscript and a superscript.

In DTS, the sentences (2.11a) and (2.11b) are given the semantic representations (2.12a) and (2.12b), respectively.

(2.12) a. $\left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right]$

$$\frac{A : \mathbf{type} \quad A \ \ true}{@_i :: A : A}\ @$$

Figure 2.4: Inference rule of @.

b. **whistle**($@_1$ :: **entity**)

The term $@_1$ :: **entity** in (2.12b) is called an *underspecified term* in DTS. The @-term plays the role of a gap to be filled by the antecedent of the pronoun in question. The form $@_i$ :: Λ is called *type annotation*, where Λ specifies the type of the underspecified term. Figure 2.4 shows the inference rule of the underspecified term, @-rule. We call the system which adds @-rule to inference rules of dependent type theory *underspecified dependent type theory (UDTT)*.

Underspecified terms are introduced by lexical entries of anaphoric expressions: in the current case, it is introduced by the pronoun *he*.[11]

$$he_{nom}; NP; @_i :: \mathbf{e}$$

Here, the subscript number is used to distinguish between @-terms introduced by different lexical items. Note that it is not the index indicating its antecedent in the example (2.11).

When two sentences, whose semantic representations are M and N, compose a discourse, the semantic representation of the whole discourse is given in terms of a Σ-type by the *progressive conjunction* (Ranta, 1994).

$$M; N \overset{def}{\equiv} \begin{bmatrix} u : M \\ N \end{bmatrix} \quad \text{where } u \notin fv(N)$$

[11] For simplicity, we omit the gender information and treat the pronoun *he* as an expression referring to an entity.

Thus, the semantic representation of the mini-discourse (2.11) is given as follows, by conjoining (2.12a) and (2.12b).

$$\left[\begin{array}{l} u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(@_1 :: \textbf{entity}) \end{array}\right] \tag{2.13}$$

The intended resolution under the current reading is shown below.

$$\left[\begin{array}{l} u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(\pi_1 u) \end{array}\right] \tag{2.14}$$

The argument of the predicate **whistle**, i.e., the place of @-term in (2.13), is to be filled by the term $\pi_1 u$. Note that the semantic representation (2.14) is equivalent to $\left(x : \textbf{entity}\right) \times (\textbf{man}(x) \times \textbf{enter}(x) \times \textbf{whistle}(x))$, which says that there is an entity that satisfies the three conditions, **man**(x), **enter**(x) and **whistle**(x).

Now the question is how to derive (2.14) from (2.13). In DTS, anaphora resolution is defined as an operation to replace underspecified terms with concrete proof terms. The process consists of *type checking*, *proof search*, and @*-elimination*. Type checking is triggered by the *felicity condition* of a sentence or a discourse, i.e., the requirement that the semantic representation of a sentence or a discourse is a *type* under the given context. Thus, for the mini-discourse to be felicitous, the following judgement must hold:

$$\mathcal{K} \vdash \begin{bmatrix} u : \begin{bmatrix} x : \textbf{entity} \\ \begin{bmatrix} \textbf{man}(x) \\ \textbf{enter}(x) \end{bmatrix} \end{bmatrix} \\ \textbf{whistle}(@_1 :: \textbf{entity}) \end{bmatrix} : \textbf{type} \tag{2.15}$$

The context $\mathcal{K}$ is called a *global context*. The global context $\mathcal{K}$ includes the basic ontological commitment (e.g., **entity** : **type**) and the arities of predicates (e.g., **man** : **entity** → **type**). It also contains ontological knowledge encoded as judgements, e.g., $john$: **entity** and $f : \left(u : \left(x : \textbf{entity}\right) \times \textbf{cat}(x)\right) \rightarrow$ **animal**$(\pi_1 u)$. Type checking follows inference rules in UDTT, namely, inference rules of dependent type theory and @-rule. The derivation tree for type checking of (2.15) is as follows.

(2.16)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\textbf{man} : \textbf{entity} \rightarrow \textbf{type}}\,con \quad \dfrac{}{x : \textbf{entity}}\,2}{\textbf{man}(x) : \textbf{type}}\,\Pi E \quad \dfrac{\dfrac{}{y : \textbf{man}(x)}\,3 \quad \dfrac{}{\textbf{enter} : \textbf{entity} \rightarrow \textbf{type}}\,con \quad \dfrac{}{x : \textbf{entity}}\,2}{\textbf{enter(x)} : \textbf{type}}\,\Pi E}{\textbf{enter}(x) : \textbf{type}}\,wk}{\begin{bmatrix} \textbf{man}(x) \\ \textbf{enter(x)} \end{bmatrix} : \textbf{type}}\,\Sigma F, 3 \quad \dfrac{}{\begin{bmatrix} x : \textbf{entity} \\ \begin{bmatrix} \textbf{man}(x) \\ \textbf{enter}(x) \end{bmatrix} \end{bmatrix}}\,1}{\begin{bmatrix} x : \textbf{entity} \\ \begin{bmatrix} \textbf{man}(x) \\ \textbf{enter}(x) \end{bmatrix} \end{bmatrix} : \textbf{type}}\,\Sigma F, 2 \quad \dfrac{\dfrac{}{\textbf{whistle} : \textbf{entity} \rightarrow \textbf{type}}\,con \quad \dfrac{\dfrac{}{\textbf{entity} : \textbf{type}}\,con \quad \begin{array}{c}\vdots \\ \textbf{entity}\ \mathit{true}\end{array}}{(@_1 :: \textbf{entity}) : \textbf{entity}}\,@}{\textbf{whistle}(@_1 :: \textbf{entity}) : \textbf{type}}\,\Pi E}{\begin{bmatrix} u : \begin{bmatrix} x : \textbf{entity} \\ \begin{bmatrix} \textbf{man}(x) \\ \textbf{enter}(x) \end{bmatrix} \end{bmatrix} \\ \textbf{whistle}(@_1 :: \textbf{entity}) \end{bmatrix} : \textbf{type}}\,\Sigma F, 1$$

The notable step is the one where the @-rule is applied. At the rightmost branch, we must show that (i) **entity** is a type, which follows from the global context, and (ii) there exists a term of **entity** under the given context. The goal to be proved for (ii) is sorted out as follows.

$$\mathcal{K},\ u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \vdash \textbf{entity}\ \textit{true} \tag{2.17}$$

Note that we can use global context $\mathcal{K}$ as well as proof term $u : (x : \textbf{entity}) \times (\textbf{man}(x) \times \textbf{enter}(x))$ to prove the judgement. The latter is the assumption that is discharged by the Σ-formation rule afterward at the bottom. Inference rules with discharging, e.g., Σ-formation and Π-formation, in general provide such a *local context*.[12] In this way, the preceding discourse's semantic representation contributes to the local context for anaphora resolution.

The judgement (2.17) launches proof search. In this case, we can prove that $\pi_1 u$: **entity** exists. Thus, we have the following derivation of UDTT, where the derivation in the rightmost branch is completed.

[12] Bekki and Mineshima (2017) defined the underspecified term as a function from a local context to the annotated type: it is defined to explicitly pass the local context to the subsequent component. In the DTS version that we adopt here, we define @-term as a term of the annotated type and consider that the local context is given as the assumptions during the type-checking. This formalism has been adopted in several studies, e.g., Tanaka et al. (2017a); Kinoshita et al. (2018); Yana et al. (2019).

(2.18)

$$\dfrac{\dfrac{\begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix} : \mathbf{type} \quad \dfrac{\dfrac{}{\mathbf{whistle} : \mathbf{entity} \to \mathbf{type}}\,con \quad \dfrac{\dfrac{}{\mathbf{entity} : \mathbf{type}}\,con \quad \dfrac{\dfrac{}{u : \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix}}\,1}{\pi_1 u : \mathbf{entity}}\,\Sigma E}{@_1 :: \mathbf{entity} : \mathbf{entity}}\,@}{\mathbf{whistle}(@_1 :: \mathbf{entity}) : \mathbf{type}}\,\Pi E}{\begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix} \\ \mathbf{whistle}(@_1 :: \mathbf{entity}) \end{bmatrix} : \mathbf{type}}\,\Sigma F, 1$$

We next substitute $@_1$ with the obtained proof term and eliminate every occurrence of $@_1$ in the derivation. This process is called @-elimination. By substituting $\pi_1 u$ for $@_1$, we finally obtain the fully-specified derivation.

(2.19)

$$\dfrac{\dfrac{\begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix} : \mathbf{type} \quad \dfrac{\dfrac{}{\mathbf{whistle} : \mathbf{entity} \to \mathbf{type}}\,con \quad \dfrac{\dfrac{}{u : \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix}}\,1}{\pi_1 u : \mathbf{entity}}\,\Sigma E}{\mathbf{whistle}(\pi_1 u) : \mathbf{type}}\,\Pi E}{\begin{bmatrix} u : \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{bmatrix} \end{bmatrix} \\ \mathbf{whistle}(\pi_1 u) \end{bmatrix} : \mathbf{type}}\,\Sigma F, 1$$

In this way, we can obtain the fully-specified semantic representation in (2.14) which substitute $\pi_1 u$ for @-term.[13]

[13] DTS can account for various factors that select plausible antecedents among entities appearing in the discourse. For instance, the notion of accessibility is obtained for

The final semantic representation in (2.14) shows how Σ-types capture the externally-dynamic aspects of existential sentences. The proof term u introduced by the Σ-type, which corresponds to the first sentence (2.11a), is a structured object (i.e., a tuple). The point is again that even if the variable x itself is no longer accessible from the argument position of **whistle**, one can access the term by taking the first projection of u.

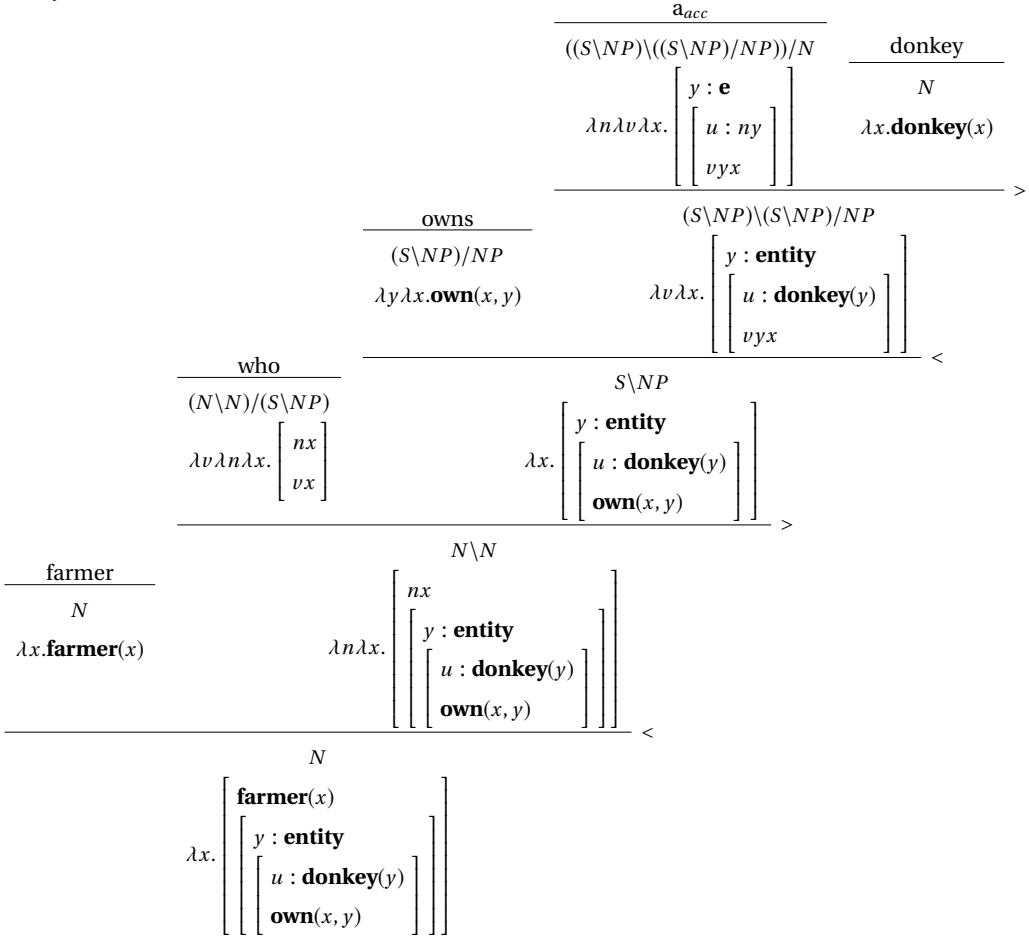
Anaphora resolution in the donkey sentence (2.1), repeated here as (2.20), is also an instance of Σ-type anaphora.

(2.20) Every farmer who owns a[1] donkey beats it$_1$.

The noun phrase is composed as follows.

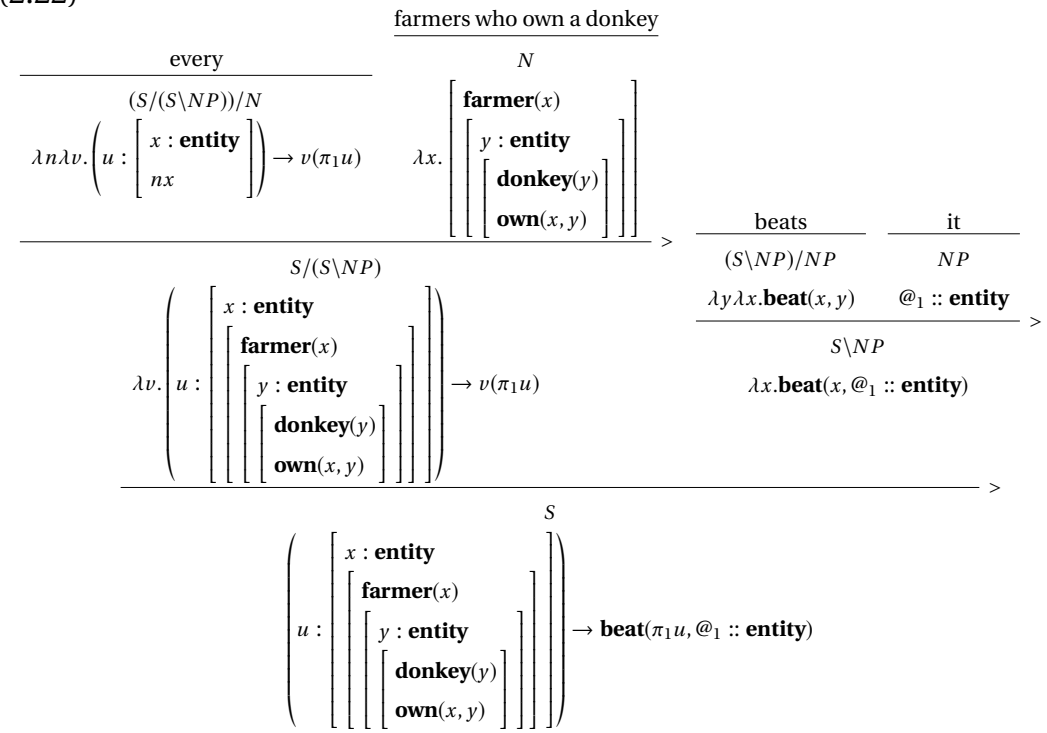
free from the structure of dependent types Σ and Π (cf. Bekki, 2014). Agreement on pronouns, such as gender, person, and number agreement, can also be accounted for by annotating the underspecified term with more fine-grained types, as in the case of presupposition (Bekki and Mineshima, 2017; Tanaka et al., 2017b).

(2.21)



The semantic representation of the whole sentence is then derived as follows.

(2.22)



In the type-checking process, we will find that the following judgement should hold for $@_1$ in order for the whole semantic representation being well-typed.

(2.23) $$\mathcal{K},\ u:\left[\begin{array}{l}x:\textbf{entity}\\ \left[\begin{array}{l}\textbf{farmer}(x)\\ \left[\begin{array}{l}y:\textbf{entity}\\ \left[\begin{array}{l}\textbf{donkey}(y)\\ \textbf{own}(x,y)\end{array}\right]\end{array}\right]\end{array}\right]\end{array}\right]\vdash \textbf{entity}\ \textit{true}$$

We can obtain $\pi_1\pi_2\pi_2 u:\textbf{entity}$ from the tuple u. Thus, we finally obtain the following fully-specified representation, where the second argument of **beat** is filled with the variable corresponding to the donkey.

$$(2.24)\quad \left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{farmer}(x) \\ \left[\begin{array}{l} y : \textbf{entity} \\ \left[\begin{array}{l} \textbf{donkey}(y) \\ \textbf{own}(x,y) \end{array} \right] \end{array} \right] \end{array} \right] \end{array} \right] \right) \rightarrow \textbf{beat}(\pi_1 u, \pi_1\pi_2\pi_2 u)$$

2.5.2 Π-type anaphora

As we have seen so far, a Σ-type is externally dynamic in that it introduces a pair as an antecedent. By contrast, Π-types capture the *externally static* property of universal quantifiers exemplified in (2.25): as the universal sentence represented by a Π-type is a function, neither the proof in the antecedent nor the consequent is directly accessible to the subsequent discourse.

(2.25) * Everybody bought a[1] car. It$_1$ is a Porsche.

One difference from the standard treatment in dynamic semantics is that a universal sentence can introduce a discourse referent. Ranta (1994) describes exactly such a case with the following example.[14]

(2.26) If you give every child a present, some child will open it.

In (2.26), the antecedent clause is analyzed as a Π-type, so it introduces a *function* as a discourse referent. This functional discourse referent can be used to give the interpretation of the pronoun *it* in the consequent clause. We call this type of anaphora Π*-type anaphora*, where a functional discourse referent is used in the subsequent discourse and contributes to resolving anaphoric expressions.

[14]This example is taken from Ranta (1994, p. 97), attributed to Lauri Karttunen in Hintikka and Carlson (1979).

A typical instance of Π-type anaphora is the so-called *quantificational subordination*, which is exhibited by (2.27) and (2.28).

(2.27) a. If every boy receives a^1 present, some boy will open it$_1$.

b. Every boy will receive a^2 present. Some boy will open it$_2$.

(2.28) a. Harvey courts a^1 girl at every convention. She$_1$ is very pretty.

b. Harvey courts a^1 girl at every convention. She$_1$ always comes to the banquet with him. The$_1$ girl is usually also very pretty.

Let us focus on a reading of (2.27a), where *every boy* takes wide scope over *a present* (henceforth, ∀–∃ reading). In its most natural reading, the pronoun *it* in the consequent clause refers to the present that the boy in question received. In other words, the pronoun receives an interpretation which depends on *some boy*. A similar reading applies to (2.27b): the pronoun *it* in the second sentence refers to the present that the boy received.

A similar observation can be made about (2.28), which was originally discussed by Karttunen (1976). Although the example is more complicated than the previous one, a similar structure seems to be involved. In (2.28a), if we force ourselves to keep the ∀–∃ reading, there is no way to establish an anaphoric link between *a girl* and the singular pronoun *she*, and hence the discourse becomes infelicitous. Thus, (2.28a) can only mean that *there is one specific girl* such that Harvey courts her at every convention and she is very pretty. In contrast, in (2.28b), the anaphoric link between *a girl* and *she* is licensed even under the ∀–∃ reading. Although the first sentence is the same as that of (2.28a), the singular pronoun *she* can refer to a girl at each convention because the quantificational adverbs such as *always* and *usually* in the subsequent discourse provide links to *every convention*.

This observation suggests that the dependency relation between objects is also an essential resource for anaphora resolution. Because a dependent interpretation is crucially involved in plural anaphora phenomena, constructing a formal mechanism to account for dependencies is one of the central issues in the literature. The standard approach is to model dependencies as sets of assignments (van den Berg, 1996a,b; Nouwen, 2003; Brasoveanu, 2008). Another approach is to model it using an extended notion of assignment functions called *parametrized sum individuals* (Krifka, 1996). However, since integrating functional relations directly into the underlying semantics is not straightforward, both approaches require substantial changes to the central notion of context to account for dependent interpretations. By contrast, DTS can account for dependency relations in terms of Π-types, which are independently motivated objects readily provided in the formal system.

Let us illustrate how Π-types encode the dependency relations in question and contribute to anaphora resolution. Consider (2.27b). Again, as a universal quantifier corresponds to a Π-type, the first sentence can be represented as follows.

$$(2.29)\quad \left(u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{array}\right]\right) \rightarrow \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{present}(y) \\ \mathbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right]$$

The term $\pi_1 u$ picks up the entity being a boy. The type represents the proposition that, for every boy, there exists a present such that the boy received it. This representation corresponds to the $\forall$–$\exists$ reading of the first sentence. Thus, a term of this type is a function that receives a pair consisting of an entity and a proof that it is a boy, and then returns a tuple that consists of an entity, a proof that it is a present, and a proof that the boy

and the present are in the receiving relation.

The second sentence is represented by the Σ-type, where the pronoun *it* is defined as an underspecified term of type **entity**. Thus, by combining the semantic representation of the two sentences in terms of progressive conjunction, (2.27b) is represented as the following Σ-type.

$$\left[\begin{array}{l} f : \left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{boy}(x) \end{array}\right]\right) \rightarrow \left[\begin{array}{l} y : \textbf{entity} \\ \left[\begin{array}{l} \textbf{present}(y) \\ \textbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} x' : \textbf{entity} \\ \left[\begin{array}{l} \textbf{boy}(x') \\ \textbf{open}(x', @_1 :: \textbf{entity}) \end{array}\right] \end{array}\right] \end{array}\right] \tag{2.30}$$

Now, the proof term f of the first sentence, which corresponds to a dependency relation between boys and presents, exists as an antecedent of the underspecified term. In the current case, type-checking yields the following inference for $@_1$.

(2.31)

$$\mathcal{K},\ f : \left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{boy}(x) \end{array}\right]\right) \rightarrow \left[\begin{array}{l} y : \textbf{entity} \\ \left[\begin{array}{l} \textbf{present}(y) \\ \textbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right],\ x' : \textbf{entity},\ b : \textbf{boy}(x') \vdash \textbf{entity}\ \mathit{true}$$

There are three proof terms in the local context of the @-term. The term f is a proof term of the first sentence. The term x' is an entity and b is a proof that x' is a boy. The proof construction goes as follows:

(2.32)

$$\dfrac{f : \left(u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix}\right) \to \begin{bmatrix} y : \textbf{entity} \\ \begin{bmatrix} \textbf{present}(y) \\ \textbf{receive}(\pi_1 u, y) \end{bmatrix} \end{bmatrix} \quad \dfrac{x' : \textbf{entity} \quad b : \textbf{boy}(x')}{\langle x', b\rangle : \begin{bmatrix} x' : \textbf{entity} \\ \textbf{boy}(x') \end{bmatrix}}\,\Sigma I}{\dfrac{f(\langle x', b\rangle) : \begin{bmatrix} y : \textbf{entity} \\ \begin{bmatrix} \textbf{present}(y) \\ \textbf{receive}(x', y) \end{bmatrix} \end{bmatrix}}{\pi_1(f(\langle x', b\rangle)) : \textbf{entity}}\,\Sigma E}\,\Pi E$$

Here we first construct a pair $\langle x', b\rangle$, which corresponds to *some boy*. By applying this to f, we obtain $f(\langle x', b\rangle)$ corresponding to the boy's present. Thus, by taking its first projection, we obtain a term $\pi_1 f(\langle x', b\rangle)$ of type **entity**. Therefore, by replacing the @-term with the obtained term, we obtain the fully-specified semantic representation.

(2.33)
$$\begin{bmatrix} f : \left(u : \begin{bmatrix} x : \textbf{entity} \\ \textbf{boy}(x) \end{bmatrix}\right) \to \begin{bmatrix} y : \textbf{entity} \\ \begin{bmatrix} \textbf{present}(y) \\ \textbf{receive}(\pi_1 u, y) \end{bmatrix} \end{bmatrix} \\ \begin{bmatrix} x' : \textbf{entity} \\ \begin{bmatrix} \textbf{boy}(x') \\ \textbf{open}(x', \pi_1 f(\langle x', b\rangle)) \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

We can see that the second argument of **open** depends on the term x', namely, an entity corresponding to the subject of the second sentence. In this way, we can account for the dependent interpretation of the pronoun *it* in (2.27b).

One can think that proof terms have something in common with discourse referents in Discourse Representation Theory (Kamp and Reyle, 1993; Kamp et al., 2011) in that both objects are introduced by sentences and referred to afterward to resolve anaphora. As we have already seen,

however, there are at least two crucial differences. Firstly, as Ranta (1994) discussed, while discourse referents are limited to individuals without any inner structure, proof terms can have any type. Secondly, together with the DTS's anaphora resolution mechanism, proof terms can contribute to logical inference, yielding a new proof term serving as an antecedent. In other words, the antecedent of pronominal anaphora is constructed through proof search. This proof-theoretic account of anaphora resolution also applies to presupposition resolution, as shown in the next section. [15]

2.6 Analysis of presupposition

DTS gives a uniform account of anaphora resolution and presupposition resolution in a proof-theoretic setting, following the "presupposition as anaphora" paradigm proposed by van der Sandt and Geurts (1991); van der Sandt (1992). In the analysis of presupposition in DTS, presupposition triggers are represented in terms of underspecified terms, @-terms. Underspecified terms can have more complex types when they represent presupposition triggers: the annotated types can be propositions of any kind.

Let us explain some examples in the following sections. In Section 2.6.1, we show one of the most basic cases: definite description. In Section 2.6.2, we explain the possessive case, a more complicated case where an underspecified term is embedded in another underspecified term.

[15] See also Yana et al. (2019) for a detailed comparison of DTS and DRT.

2.6.1 Definite description

Our first example is the existential presupposition triggered by a definite description. Consider the sentence (2.34) with the definite subject NP.

(2.34) The man whistled.

The semantic representation of (2.34) is given as follows.

$$\text{(2.35)}\quad \textbf{whistle}\left(\pi_1\left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ \textbf{man}(x) \end{bmatrix}\right)\right)$$

The underspecified term is introduced by the presupposition trigger *the*. The lexical entry for *the* can be specified as follows.

$$\text{(2.36)}\quad \text{the; } (S/(S\backslash NP))/N;\ \lambda n.\lambda v.\, v\left(\pi_1\left(@_i :: \begin{bmatrix} x : \textbf{entity} \\ nx \end{bmatrix}\right)\right)$$

The annotated type represents the proposition *there is a man*, which is the existential presupposition triggered by the definite description *the man*.[16] Thus, the semantic representation in (2.35) is obtained by the following CCG derivation.

[16] Here we take it that the uniqueness presupposition is not part of the conventional meaning of a definite description but can be derived on pragmatic considerations along the lines of Heim (1982). Although the proof-search procedure to find an underspecified term's antecedent would become much more complicated, it is technically possible to take the uniqueness implication as part of presupposition.

(2.37)

$$\dfrac{\dfrac{\dfrac{\text{The}}{(S/(S\backslash NP))/N}}{\lambda n.\lambda v.\, v\left(\pi_1\left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ nx \end{bmatrix}\right)\right)} \quad \dfrac{\dfrac{\text{man}}{N}}{\lambda x.\, \textbf{man}(x)}}{S/(S\backslash NP)} > \quad \dfrac{\dfrac{\text{whistled}}{S\backslash NP}}{\lambda x.\, \textbf{whistle}(x)}$$

$$\lambda v.\, v\left(\pi_1\left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ \textbf{man}(x) \end{bmatrix}\right)\right) \qquad\qquad > \quad \textbf{whistle}\left(\pi_1\left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ \textbf{man}(x) \end{bmatrix}\right)\right)$$

Again, as in the case of anaphora resolution, the resolution of an underspecified term in a semantic representation amounts to checking that the semantic representation is well-typed in a given context. Thus, the resolution process is launched by the following judgement.

$$\mathcal{K} \vdash \textbf{whistle}\left(\pi_1\left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ \textbf{man}(x) \end{bmatrix}\right)\right) : \textbf{type} \tag{2.38}$$

Assuming that **whistle** : **entity** $\rightarrow$ **type** is in the global context $\mathcal{K}$, one has to prove the following in order for the semantic representation in question to be well-typed.

$$\mathcal{K} \vdash \begin{bmatrix} x : \textbf{entity} \\ \textbf{man}(x) \end{bmatrix} \mathit{true} \tag{2.39}$$

In other words, if one assumes that (2.34) is a felicitous utterance, the proposition that *there is a man* must be true. This requirement corresponds to the existential presupposition of (2.34). Thus, at the final stage of presupposition resolution, a proof search is carried out to prove (2.39), and the underspecified term $@_1$ is replaced by the constructed term. Such

a proof construction is possible when, for instance, *the man* appears in contexts as shown in (2.40a, b).

(2.40) a. A man entered. The man whistled.

b. If a man entered, the man will whistle.

In both cases, *the man* can be understood as referring to *a man* previously mentioned. In general, if S' entails the presuppositions of S, constructions such as S' and S and If S' then S do not inherit the presuppositions of S. In such a case, it is said that the presupposition is *filtered*.[17]

In DTS, examples such as (2.40a, b) can be handled in the following way. The semantic representation for the first sentence of (2.40a) is already given in (2.6). Thus, by progressive conjunction with (2.35), we obtain the following semantic representation of the whole discourse.

$$\left[\begin{array}{l} v : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{array}\right] \end{array}\right] \\ \mathbf{whistle}\left(\pi_1\left(@_1 :: \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right]\right)\right) \end{array}\right] \tag{2.41}$$

According to the type checking, we can see that what is required for the representation (2.41) to be well-typed is to prove the following judgement.

$$\mathcal{K},\ v : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{enter}(x) \end{array}\right] \end{array}\right] \vdash \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{man}(x) \end{array}\right] \ \mathit{true} \tag{2.42}$$

In this case, a proof of the proposition that there is a man can be obtained from the context, namely, from the semantic representation of the first

[17] When such a filtration does not occur, the entire sentence can have a presupposition resolved by the information in the background knowledge (the global context).

sentence: one can find $\langle \pi_1 v, \pi_1 \pi_2 v \rangle$ that can replace the @-term. By replacing the $@_1$ in the representation (2.41) with the constructed term, we can eventually obtain the following semantic representation for (2.40a), which captures the intended reading.

(2.43)
$$\left[\begin{array}{l} v : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(\pi_1 v) \end{array}\right]$$

Another well-known characteristic property of a presupposition is that it *projects* out of embedded contexts such as negation and the antecedent of a conditional. Thus, not only the positive sentence (2.34) but also the negated sentence (2.44a) and the antecedent of a conditional (2.44b) imply that there is a man.

(2.44) a. The man didn't whistle. NEGATION

b. If the man whistle, Susan will be surprised. CONDITIONAL

In DTS, (2.44a, b) can be given the following semantic representations.

(2.45) a. $\neg \textbf{whistle}\left(\pi_1\left(@_1 :: \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{man}(x) \end{array}\right]\right)\right)$

b. $\textbf{whistle}\left(\pi_1\left(@_1 :: \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{man}(x) \end{array}\right]\right)\right) \rightarrow \textbf{surprise}\,(\mathsf{susan})$

For the semantic representations (2.45a) and (2.45b) to be well-typed, it is required to find a proof term for the proposition that there is a man. Thus, to prove that (2.45b) has type **type**, one has to prove that the antecedent is of type **type**. Since the antecedent in (2.45b) corresponds to the proposition that the man whistled, this yields the derivation that contains the type checking process of (2.38) as a sub-derivation. Accordingly,

it is correctly predicted that (2.44b) has the same existential presupposition as the simple sentence in (2.34). Note that in dependent type theory, the negation $\neg A$ is defined using the implication $A \to \bot$, which in turn is a degenerate form of a Π-type. Thus, the same explanation applies to the case of negation in (2.44a) as well. In this way, we can explain basic projection patterns of presuppositions within the framework of DTS.[18]

2.6.2 Possessives

Next, let us explain an analysis of the presupposition associated with possessive NPs. Consider the sentence (2.46), whose semantic representation is given in (2.47).

(2.46) His sister left.

(2.47) $$\textbf{leave}\left(\pi_1\left(@_1 :: \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{sisterOf}(x, @_2 :: \textbf{entity}) \end{array}\right]\right)\right)$$

Here, underspecified terms that correspond to distinct proof terms are indexed with different natural numbers. In (2.47), $@_2$ is embedded in the annotated type of $@_1$. The derivations of initial underspecified semantic representations can be formalized in a fully compositional way. The lexical entry for *sister* and *his* can be specified as follows, where we use the CCG category RN for relational nouns.

$$\text{sister}; RN; \lambda y \lambda x.\textbf{sisterOf}(x, y)$$

$$\text{his}; NP/RN; \lambda R.\pi_1\left(@_i :: \left[\begin{array}{l} x : \textbf{entity} \\ R(x, @_j :: \textbf{entity}) \end{array}\right]\right)$$

[18] Bekki and Satoh (2015) provide a definition of a decidable fragment of dependent type theory with an underspecified term and formulate its type-checking algorithm. They also provide an implementation of the algorithm.

Note that we assume that in initial underspecified semantic representations, each occurrence of @ is assigned a mutually distinct index. The embedded $@_j$ corresponds to the anaphoric property of the pronoun, while $@_i$ corresponds to the possessive presupposition.

To prove that the semantic representation in (2.47) is a type under the global context $\mathcal{K}$, the following judgement (2.48) should hold for $@_1$.

$$\mathcal{K} \vdash \left(@_1 :: \begin{bmatrix} x : \textbf{entity} \\ \textbf{sisterOf}(x, @_2 :: \textbf{entity}) \end{bmatrix} \right) : \begin{bmatrix} x : \textbf{entity} \\ \textbf{sisterOf}(x, @_2 :: \textbf{entity}) \end{bmatrix} \tag{2.48}$$

By applying the @-rule to (2.48), we are required to prove that the following type is well-typed.

$$\mathcal{K} \vdash \begin{bmatrix} x : \textbf{entity} \\ \textbf{sisterOf}(x, @_2 :: \textbf{entity}) \end{bmatrix} : \textbf{type} \tag{2.49}$$

This requirement leads us to the next judgement (2.50) for the embedded underspecified term $@_2$. We first need to resolve $@_2$ in (2.50);

$$\mathcal{K}, x : \textbf{entity} \vdash @_2 :: \textbf{entity} : \textbf{entity} \tag{2.50}$$

This triggers proof search to find a term of **entity** for $@_2$.

Suppose that the proof search finds a contextually salient man, John. By substituting $\mathsf{john} : \textbf{entity}$ in $\mathcal{K}$ for $@_2$, we are next going to prove the following judgement for $@_1$.

$$\mathcal{K} \vdash \begin{bmatrix} x : \textbf{entity} \\ \textbf{sisterOf}(x, \mathsf{john}) \end{bmatrix} \textit{true} \tag{2.51}$$

Then, in this case, the presupposition of the sentence (2.46) is predicted to be *John has a sister*.

2.7 Anaphora resolution and inference

We have shown that DTS uniformly treats anaphora and presupposition resolution as a proof search for resolving an underspecified term. This is a powerful mechanism that can apply to anaphora and presupposition resolution that involves general inference.

By the example of quantificational subordination in Section 2.5.2, repeated here as (2.52), we have shown that one can obtain the antecedent term by applying function with its argument.

(2.52) Every boy will receive a^1 present. Some boy will open it$_1$.

In (2.52), the subject of the antecedent and the consequent share the same noun phrase *boy*. The reference to a dependency relation is possible more generally, when a semantic link between the restrictor of the universal quantifier and the subject of a subsequent sentence can be established.

(2.53) a. Every boy will receive a^1 present. Every young boy will open it$_1$.

b. Every boy will receive a^1 present. John will open it$_1$.

In (2.53a), *young boy* is a subset of *boy*. The consequent can be understood to mean that every young boy will open the present he received (cf. van den Berg, 1996b). There is no explicit link in the case of (2.53b), but if we have the background information that John is a boy, i.e., the information that links *John* to *boy*, the sentence can mean that John will open the present he received. On the other hand, the dependent interpretation of the pronoun in question is not possible if it is difficult to establish a link between the two NPs. The following examples demonstrate this contrast.

(2.54) a. Every man will receive a^1 present. Some wife will open it$_1$.

b. Every man will receive a[1] present. *Some woman will open it$_1$.

In (2.54a), since it is relatively easy to find a relation between *man* and *wife*, the second sentence can be understood to mean that some man's wife will open the present he received. This contrasts with (2.54b), where a dependent reading is not possible unless a strong relation between *man* and *woman* is provided by the context.

Let us sketch how DTS can account for such cases. In (2.53a), both the first and the second sentences can be represented in terms of Π-types. Thus, we can give the following semantic representation to the discourse.

(2.55)
$$\left[\begin{array}{l} f : \left(u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{array}\right]\right) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{present}(y) \\ \mathbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right] \\ \left(t : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{boy}(x) \\ \mathbf{young}(x) \end{array}\right] \end{array}\right]\right) \to \mathbf{open}(\pi_1 t, @_1 :: \mathbf{entity}) \end{array}\right]$$

The assumptions we can use in proof search of $@_1$ are terms f and t. We need a proof of $(x : \mathbf{entity}) \times \mathbf{boy}(x)$ in order to access to the consequent of f. As we can obtain such a proof $\langle \pi_1 t, \pi_1 \pi_2 t \rangle$ from t, we eventually obtain a term $\pi_1 f(\langle \pi_1 t, \pi_1 \pi_2 t \rangle)$, which corresponds to the present dependent on each young boy, $\pi_1 t$. Similarly, (2.53b) is represented as follows.

(2.56)
$$\left[\begin{array}{l} f : \left(u : \left[\begin{array}{l} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{array}\right]\right) \to \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{present}(y) \\ \mathbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right] \\ \mathbf{open}(\mathsf{john}, @_1 :: \mathbf{entity}) \end{array}\right]$$

To find a semantic link between *John* and *boy*, one needs the background

knowledge that John is a boy. If the global context $\mathcal{K}$ supplies the knowledge k_j : **boy**(john), one can construct the following term.

(2.57) $(\mathsf{john}, k_j) : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{boy}(x) \end{bmatrix}$

Again, this term can serve as an argument to the function f.

When the relation between the universal quantifier's restrictor and the subsequent discourse's subject is not clear, then the procedure only fails to find a proof term. For instance, in the case of (2.54b), repeated here as (2.58a), there exists neither an explicit link nor an implicit link between men and women.

(2.58) a. Every man will receive a[1] present. *Some woman will open it$_1$.

b. $\left[f : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{man}(x) \end{bmatrix} \right) \to \begin{bmatrix} y : \mathbf{entity} \\ \begin{bmatrix} \mathbf{present}(y) \\ \mathbf{receive}(\pi_1 u, y) \end{bmatrix} \end{bmatrix} \right.$
$\left. \begin{bmatrix} z : \mathbf{entity} \\ \begin{bmatrix} \mathbf{woman}(z) \\ \mathbf{open}(z, @_1 :: \mathbf{entity}) \end{bmatrix} \end{bmatrix} \right]$

Again, one needs to apply an argument to the function f to construct a proof of the present received by some man. Thus, unless some relation that bridges men and women is available in the global context, there is no way to obtain the required proof term from the available proof terms.

Another case that exhibits the effectiveness of inference is the so-called *bridging* inference (Clark, 1975).

(2.59) John bought a car. He checked the motor.

Here, although the second sentence contains the definite description *the*

motor, there is no overt antecedent to resolve the presupposition. As hearers have a general background knowledge that *every car has a motor*, however, there are no difficulties in interpreting this sentence: they can easily infer that there is a motor from the information that there is a car.

Compared to the analyses in standard dynamic semantics such as DRT (van der Sandt, 1992; Geurts, 1999; Kamp et al., 2011), it is straightforward for DTS to handle the bridging inference. DTS provides an analysis that exactly follows the intuition described above. By using both the implicit information in the background knowledge and explicit information in the given sentences, one can construct the proof term about the existence of a motor. See Bekki and Mineshima (2017) for more details about the analysis of bridging inference in DTS.

The idea that the reference of an anaphoric expression can be constructed via inference originates from Krahmer and Piwek (1999). It should be noted here, however, that a naive implementation of this idea leads to a problem similar to *formal link* between a pronoun and an antecedent in the context of E-type approaches to pronouns (Heim, 1990); that is, the theory over-generates in the case of (2.61).

(2.60) A man has a wife. She is sitting next to him.

(2.61) * A man is married. She is sitting next to him.

In contrast to (2.60), the pronoun *she* in (2.61) cannot anaphorically refer to the man's wife, because there is no NP antecedent. If there is the knowledge that every married man has a wife, however, the referent corresponding to the man's wife can be constructed via inference.

Since it is beyond the scope of this thesis to discuss this issue further, for present purposes, we will assume that any inference is possible for given implicit and explicit information, though some cases of proof search

might be very costly and not preferred.

2.8 Entailment and meaning

Once a semantic representation is fully-specified by eliminating every occurrence of @-term, namely, the sentence's interpretation has been fixed, the logical consequence of the given semantic representation becomes clear. We can verify the appropriateness of a semantic representation in DTS by seeing the entailment patterns it allows.

In model-theoretic semantics, the correctness of semantic representation of a sentence predicted by a theory is checked in terms of their truth condition: when $\mathcal{R}'$ is a semantic representation of natural language sentence $\mathcal{R}$, then the truth condition of $\mathcal{R}'$ must agree with the truth condition of $\mathcal{R}$. On the other hand, in our proof-theoretic setting, the correctness of semantic representation of a sentence is checked in terms of entailment relations: if sentence $\mathcal{R}_1$ entails $\mathcal{R}_2$, we should be able to prove that $\mathcal{R}'_2$ is derivable from $\mathcal{R}'_1$ in dependent type theory.

For instance, the following entailment relations hold between natural language sentences.

(2.62) A^1 man entered. He_1 whistled. $\Rightarrow$ A man entered.

(2.63) A^1 man entered. He_1 whistled. $\Rightarrow$ A man whistled.

(2.64) Every boy received a present. John is a boy. $\Rightarrow$ John received a present.

Note that (2.62) holds no matter what object the pronoun *he* refers to, while (2.63) holds only under the reading where *he* refers to *a man* in the first sentence.

DTS can provide a semantic representation for the sentence on each side. If those semantic presentations correctly capture the meaning of original sentences, then the formula of the premise must entail the formula of the conclusion. Thus, the following judgement (2.65)-(2.67) must hold for (2.62)-(2.64), respectively.

$$(2.65)\quad \mathcal{K},\ p:\left[\begin{array}{l} u:\left[\begin{array}{l} x:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{man}(x)\\ \mathbf{enter}(x)\end{array}\right]\end{array}\right] \\ \mathbf{whistle}(\pi_1 u)\end{array}\right] \vdash \left[\begin{array}{l} x:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{man}(x)\\ \mathbf{enter}(x)\end{array}\right]\end{array}\right] \mathit{true}$$

$$(2.66)\quad \mathcal{K},\ p:\left[\begin{array}{l} u:\left[\begin{array}{l} x:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{man}(x)\\ \mathbf{enter}(x)\end{array}\right]\end{array}\right] \\ \mathbf{whistle}(\pi_1 u)\end{array}\right] \vdash \left[\begin{array}{l} x:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{man}(x)\\ \mathbf{whistle}(x)\end{array}\right]\end{array}\right] \mathit{true}$$

$$(2.67)\quad \mathcal{K},\ p_1:\left(u:\left[\begin{array}{l} x:\mathbf{entity} \\ \mathbf{boy}(x)\end{array}\right]\right)\rightarrow \left[\begin{array}{l} y:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{present}(y)\\ \mathbf{receive}(\pi_1 u, y)\end{array}\right]\end{array}\right],\ p_2:\mathbf{boy}(\mathsf{john}) \vdash \left[\begin{array}{l} y:\mathbf{entity} \\ \left[\begin{array}{l}\mathbf{present}(y)\\ \mathbf{receive}(\mathsf{john}, y)\end{array}\right]\end{array}\right] \mathit{true}$$

The proof of (2.65) is trivial: by taking its first projection, one can obtain the proof. (2.66) are proved in (2.68).

(2.68)

$$\dfrac{\dfrac{\dfrac{p : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(\pi_1 u) \end{array}\right]}{\pi_1 p : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right]}\,\Sigma E}{\pi_1\pi_1 p : \textbf{entity}}\,\Sigma E \qquad \dfrac{\dfrac{\dfrac{p : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(\pi_1 u) \end{array}\right]}{\pi_1 p : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right]}\,\Sigma E}{\pi_2\pi_1 p : \left[\begin{array}{l} \textbf{man}(\pi_1 p) \\ \textbf{enter}(\pi_1 p) \end{array}\right]}\,\Sigma E \qquad \dfrac{p : \left[\begin{array}{l} u : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{enter}(x) \end{array}\right] \end{array}\right] \\ \textbf{whistle}(\pi_1 u) \end{array}\right]}{\pi_2 p : \textbf{whistle}(\pi_1 p)}\,\Sigma E}{\langle \pi_1\pi_2\pi_1 p, \pi_2 p\rangle : \left[\begin{array}{l} \textbf{man}(\pi_1 p) \\ \textbf{whistle}(\pi_1 p) \end{array}\right]}\,\Sigma I}{\langle \pi_1\pi_1 p, \langle \pi_1\pi_2\pi_1 p, \pi_2 p\rangle\rangle : \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} \textbf{man}(x) \\ \textbf{whistle}(x) \end{array}\right] \end{array}\right]}$$

In both cases, the proofs are obtained from the premise p by extracting proofs from its part and reconstructing them. (2.67) is proved as follows.

(2.69)

$$\dfrac{p_1 : \left(u : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{boy}(x) \end{array}\right]\right) \to \left[\begin{array}{l} y : \textbf{entity} \\ \left[\begin{array}{l} \textbf{present}(y) \\ \textbf{receive}(\pi_1 u, y) \end{array}\right] \end{array}\right] \qquad \dfrac{\overline{\mathsf{john} : \textbf{entity}} \quad p_2 : \textbf{boy}(john)}{\langle \mathsf{john}, p_2\rangle : \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{boy}(x) \end{array}\right]}\,\Sigma I}{p_1(\langle \mathsf{john}, p_2\rangle) : \left[\begin{array}{l} y : \textbf{entity} \\ \left[\begin{array}{l} \textbf{present}(y) \\ \textbf{receive}(\pi_1(\langle \mathsf{john}, p_2\rangle), y) \end{array}\right] \end{array}\right]}\,\Pi I$$

Here we can understand that $\mathsf{john} : \textbf{entity}$ is in the global context, because $\textbf{boy}(\mathsf{john})$ is well-typed when *John is a boy* is a felicitous utterance. Thus, as $\pi_1(\langle \mathsf{john}, p_2\rangle)$ is reduced to john, we obtain the proof of the conclusion of (2.67).

In this way, we can show that our present analysis accounts for the empirical data shown in (2.62), (2.63), and (2.64). For more information on the *inference-as-tests paradigm* in DTS, see Bekki and Mineshima (2017).

DTS has been applied to various linguistic phenomena, providing a proof-theoretic account of natural language semantics: modal subordination (Tanaka et al., 2015), conventional implicatures (Bekki and McCready, 2015), honorification (Watanabe et al., 2014), factive presupposition (Tanaka et al., 2017b), selectional restriction (Kinoshita et al., 2017), coercion (Kinoshita et al., 2018), paycheck anaphora (Tanaka et al., 2018), interpretive parallelism (Kubota and Levine, 2015; Kubota et al., 2019), and questions (Watanabe et al., 2019). This thesis aims to give an analysis of natural language quantifiers in DTS and investigate how it can account for the linguistic and logical aspects of quantifiers.

Chapter 3

Quantifier in Natural Language

In the previous chapter, we have shown that Π-types and Σ-types correspond to universal and existential quantifiers. The meaning of *every* and *some* were given in terms of Π and Σ. The question is how other natural language quantifiers such as *most* are represented in the framework.

Before providing our analysis in the next chapter, this chapter summarizes natural language quantifiers' behavior and how the existing approaches account for them.

3.1 Quantification and determiners

Quantifiers have been studied extensively in both logic and linguistics.[1] For instance, first order logic has a universal quantifier $\forall$ and an existential quantifier $\exists$ quantifying over objects in the domain (universe): $\forall x.\phi(x)$ means *for each* e *in the domain,* $\phi(e)$ *holds* and $\exists x.\phi(x)$ means *there is some* e *in the domain such that* $\phi(e)$ *holds.* These logical expressions can be ap-

[1] Peters and Westerståhl (2006) provides an extensive survey of the studies in logic and linguistics. See also Szabolcsi (2010) for a helpful survey in linguistics.

plied to represent the statements *every A is B* and *some A is B* in the following form.

(3.1) $\forall x.(A(x) \rightarrow B(x))$

(3.2) $\exists x.(A(x) \wedge B(x))$

Generalized quantifier theory (Mostowski, 1957; Lindström, 1966) generalizes the notion of quantifier and treats universal and existential quantifiers in a unified way as part of a number of other quantifiers. Generalized quantifiers (GQs) are defined as describing a property of subsets of the domain. Universal quantifiers and existential quantifiers are reinterpreted as expressions describing the properties of a subset. Now, $\phi(x)$ is regarded as a subset of the domain that satisfies ϕ. $\forall$ is then interpreted as the subset is equal to the domain. Similarly, $\exists$ is describing that the subset is non-empty. Quantifiers such as $\forall$ and $\exists$, which take one subset and describe its property, are said to be of type $\langle 1 \rangle$. Statements *every A is B* and *some A is B* are expressed as follows with quantifiers Q_{every} and Q_{some} taking two sets.

(3.3) $Q_{every}(A, B) \Leftrightarrow A \subseteq B$

(3.4) $Q_{some}(A, B) \Leftrightarrow A \cap B \neq \emptyset$

A quantifier taking two subsets and describing their relation is said to be of type $\langle 1, 1 \rangle$.

In natural language, a determiner is naturally taken as a type $\langle 1, 1 \rangle$ quantifier, as it denotes the binary relations between two sets corresponding to a noun and a verb phrase (Barwise and Cooper, 1981). Since a determiner and noun first form a constituent and become a noun phrase, the noun phrase is considered to be a type $\langle 1 \rangle$ quantifier.

In this thesis, we concentrate on determiners as the representative case

of quantifiers in natural language. In the sentence of the form $[Det\ N]\ VP$ where Det is a quantifier, the noun N which constitutes a type $\langle 1 \rangle$ quantifier together with the quantifier Det is called *restrictor*. The verb phrase VP is called *(nuclear) scope*.[2]

We will pay particular attention to the quantifier *most*. It is a typical proportional quantifier that cannot be expressed by a first-order quantifier. It is impossible to express the statement *Most A are B* in the form of $\mathbf{M}x(A\#B)$, where $\mathbf{M}$ is the first-order quantifier that means *Most things are* ϕ (Barwise and Cooper, 1981; Szabolcsi, 2010). It is an instance of quantifiers that exhibit quantifying over the restrictor is essential to natural language quantifiers.

The quantifier *most* is often treated as truth-conditionally equivalent to *more than half*. Thus, the statement *Most A are B* is expressed as follows with several variation.

(3.5) a. $|A \cap B| > \frac{|A|}{2}$

b. $|A \cap B| > |A - B|$

Hackl (2009) argues that even the (3.5a, b) are truth-conditionally equivalent, (3.5b) better expresses the meaning of *most* from the viewpoint of their verification procedures. Hunter et al. (2008) discussed more fine variations on (3.5b). In the next chapter, we will consider both of these

[2] The quantification of natural language appears also in other linguistic expressions. For instance, quantificational adverbs such as *always* and *usually* correlates with *all* and *most* (Lewis, 1975). As quantificational adverbs involve quantification over tuples rather than individuals, they are classified as *polyadic quantifiers* rather than quantifiers representing the relationship between subsets of the domain (Peters and Westerståhl, 2006). Kamp et al. (2011) analyzes the quantificational adverb *mostly* by extending their analysis of the determiner *most*.

Quantifiers	Relations between sets
$[\![\text{every}]\!](A)(B)$	$A \subseteq B$
$[\![\text{some}]\!](A)(B)$	$A \cap B \neq \emptyset$
$[\![\text{no}]\!](A)(B)$	$A \cap B = \emptyset$
$[\![\text{most}]\!](A)(B)$	$\lvert A \cap B\rvert > \frac{\lvert A\rvert}{2}$
	$\lvert A \cap B\rvert > \lvert A - B\rvert$
$[\![\text{at least } n]\!](A)(B)$	$\lvert A \cap B\rvert \geq n$
$[\![\text{at most } n]\!](A)(B)$	$\lvert A \cap B\rvert \leq n$
$[\![\text{exactly } n]\!](A)(B)$	$\lvert A \cap B\rvert = n$

Table 3.1: Quantifiers and their definition as a relation between sets.

definition and discuss their difference in our setting.[3]

Table 3.1 shows quantifiers we consider in this thesis and their basic meanings represented by the relations between sets.

Some quantifiers are known to have a presupposition that its restrictor is non-empty. Such quantifiers are called *strong determiners*, while the others are called *weak determiners*. The distinction between weak and strong determiners originates from Milsark (1977). The quantifiers that can appear in a *there be* sentence are judged as weak, and those who cannot are strong.[4]

(3.6) a. There are some/no/at least three/at most three/exactly three cats.

[3] We also restrict ourselves to the case where the domain of quantification is finite. For the infinite case, see Topal and Çevik (2020), who proposed a semantics using natural density to deal with the quantify-related statements regarding infinite sets.

[4] We use '$*$' to indicate an infelicitous utterance/interpretation.

b. * There are every/most cats.

The quantifiers in (3.6a) are classified as weak determiners and (3.6b) are as strong determiners.

3.2 Intra-sentential anaphora

In the previous chapter, we discussed the donkey sentence (3.7) with the quantifier *every*. The construction of a donkey sentence is well-studied also in relation to other quantifiers as in (3.8) and (3.9).

(3.7) Every farmer who owns a^1 donkey beats it$_1$.

(3.8) Most farmers who own a^1 donkey beat it$_1$.

(3.9) No farmer who owns a^1 donkey beats it$_1$.

In the sentence (3.7), the pronoun *it* in the nuclear scope refers to the indefinite *a donkey* in the restrictor. Now, from the generalized-quantifier viewpoint, the restrictor and the nuclear scope are denoting sets. However, there is an additional complexity in that an anaphoric dependency exists between the nuclear scope and the restrictor. In this case, the two sets related by the quantifier *every* are not independent of each other: the set corresponding to the nuclear scope is defined only after the pronoun is resolved. As the quantifier *every* lets the restrictor accessible to the scope, it is said to be *internally dynamic*. The same applies to *most*, *no*, and other quantifiers in the Table 3.1.

It has been recognized in the literature that there are two readings in the donkey sentence above: *existential reading* and *universal reading*.[5]

[5]There has been some controversy over whether the singular donkey pronoun *it* carries uniqueness presupposition or not (Heim, 1982; Kadmon, 1990; Heim, 1990). The

(3.10) EXISTENTIAL READING (WEAK READING)
Every farmer who owns a donkey beats *at least one* donkey he owns.

(3.11) UNIVERSAL READING (STRONG READING)
Every farmer who owns a donkey beats *every* donkey he owns.

In the early discourse analysis in classical DRT, the theory predicted that (3.7)'s truth condition corresponds to (3.11). In other words, the theory predicted that the sentence is false in a situation where one of the farmers owns two donkeys and beat only one of them while all the other farmers consistently beat all the donkeys they own. However, some speakers hesitate to judge (3.7) is false in such a situation (Geurts (2002), attributed to Parsons (1978)). What is more, the interpretation of a universal donkey sentence seems to be not merely ambiguous: although the sentence like (3.12a) is preferred to get universal reading, its variant (3.12b) is preferred to get existential reading.

(3.12) a. Every guest who had a^1 credit card kept it$_1$ in his wallet.
b. Every guest who had a^1 credit card used it$_1$ to pay his hotel bill.

These observations have led to the semantic theories that can explain existential reading as well.

As is discussed in the literature, there are a variety of factors that determine the preference of those two interpretations, including the choice of quantifiers (Kanazawa, 1994), the content of the predicate (Yoon, 1996), and perhaps more general background assumptions.

discussion of existential/universal reading is based on the view that there is no systematic uniqueness presupposition associated with the pronoun (Chierchia, 1992).

Regarding the choice of quantifiers, Kanazawa (1994) examines the relations between existential/universal reading and the monotonicity inference associated with quantifiers. According to the summary of the data he observed, quantifiers are roughly categorized into three groups with respect to the two readings.

- existential reading only: weak determiners (e.g., *some, at least n, no*)
- universal reading preferred (but may have existential reading): strong determiners except *most* (e.g., *every*)
- Both reading: *most*

However, as Kanazawa himself pointed out, there exists exception in the above categorization.

(3.13) No man who had a[1] credit card failed to use it$_1$.

(3.14) No man with an[1] umbrella leaves it$_1$ home on a day like this.

Although *no* is a weak determiner, these examples seems to get universal reading on their most natural reading. He also reports that the case of *most* is relatively not clear.[6] It is still a matter of debate what factors affect the preferred reading (Yoon, 1994; Geurts, 2002; Tałasiewicz, 2018) and how the two readings can be captured in the semantics (Kanazawa, 2001; King, 2004) and pragmatics (Champollion et al., 2019).

[6] Brasoveanu (2008) pointed out that a single quantifier can cause both of the two readings with respect to the multiple instances of singular donkey pronouns.

(i) Every person who buys a[1] book on amazon.com and has a[2] credit card uses it$_2$ to pay for it$_1$.

In the natural reading of this sentence, it_2 receives the existential reading while it_1 receives the universal reading.

In the next chapter, we provide an analysis that account for basically only the existential reading of quantifiers including *most*, because existential reading is logically weaker for the quantifiers other than *no*. For the quantifier *no* and *every*, we provide also its universal reading.

3.3 Inter-sentential anaphora

To better understand the meaning of quantifiers as well as their contribution to the contexts for anaphora resolution, let us consider their external behavior in discourses.

While quantifiers such as *every* and *most* are internally dynamic, they are *externally static*: the following examples show that neither their restrictor nor the nuclear scope is accessible to the subsequent discourse, in contrast to the determiner *a*.

(3.15) a. A^1 farmer who owns a^2 donkey beats it_2.
He_1 treats it_2 badly.

b. $Every^1$ farmer who owns a^2 donkey beats it_2.
$^*He_1$ treats it_2 badly.

c. $Most^1$ farmers who own a^2 donkey beat it_2.
$^*He_1$ treats it_2 badly.

The situation, however, differs when the subsequent sentence involves a plural pronoun. The following examples exhibit that a plural pronoun can refer to the restrictor as well as the nuclear scope.

(3.16) a. $Every^1$ farmer who owns a^2 donkey beats it_2.
$They_1$ treat $them_2$ badly.

b. Most[1] farmers who own a[2] donkey beat it$_2$.
They$_1$ treat them$_2$ badly.

Note that, in both (3.16a, b), the first plural pronoun *they* in the subsequent sentence refers to *a set of all farmers that own a donkey and beat it*. That is, the pronoun takes the largest set satisfying both restrictor and scope as its antecedent (Evans, 1980).

The second plural pronoun *them* in the subsequent sentence of (3.16a, b) can be also singular under the reading where *they* in the subject position takes wider scope over *it*.

(3.17) a. Every[1] farmer who owns a[2] donkey beats it$_2$.
They$_1$ treat it$_2$ badly.

b. Most[1] farmers who own a[2] donkey beats it$_2$.
They$_1$ treat it$_2$ badly.

It is not only the existence of a plural pronoun that licenses anaphoric link to the universal quantifier's nuclear scope. Here we repeat the quantificational subordination cases we discussed in the previous chapter.

(3.18) Every boy will receive a[1] present. Some boy will open it$_1$.

(3.19) Harvey courts a[1] girl at every convention.
She$_1$ always comes to the banquet with him.
The$_1$ girl is usually very pretty.

Even when (3.18) receives ∀–∃ reading, the pronoun *it* in the second sentence can refer to *a present* in the first sentence. In this case, the pronoun is referring to the present that the boy in the second sentence receives. A similar observation applies to (3.19). Although it involves quantificational adverbs *always* and *usually*, which we will not discuss in detail, the

example suggests that the nuclear scope of the universal quantifier is accessible even in the absence of a plural pronoun.[7]

Note that plural pronoun *they* can refer to different kind of sets: *reference set, maximal set,* and *compliment set* (Nouwen, 2003).

(3.20) Few senators admire Kennedy.
They are very junior.

(3.21) Few senators admire Kennedy.
Most of them prefer Carter.

(3.22) Few senators admire Kennedy.
They admire Carter instead.

In (3.20), *they* refers to a set of senators who admire Kennedy (reference set). In (3.21), *them* refers to a set of senators, which is the restrictor set (maximal set). In (3.22), *they* refers to a set of senators that do not admire Kennedy (complement set). While anaphoric link to the reference set is robust, the other two are not generally available.

(3.23) Two senators who admire Kennedy bullied Carter. They are very junior.

(3.24) * Most senators admire Carter. They admire Kennedy instead.

The example (3.23) cannot mean that the senators who admire Kennedy are in general junior, which means the pronoun cannot refer to the max-

[7] It is known that a singular pronoun can refer even to the restrictor in some special cases as exemplified below (taken from Roberts (1989), attributed to Barbara Partee).

(i) Each[1] degree candidate walked to the stage.
He$_1$ took his$_1$ diploma from the Dean and returned to his seat.

Roberts (1989) called this phenomenon *telescoping* and explained that this anaphoric link becomes possible due to some sort of narrative continuity in the discourse.

imal set. In (3.24), the pronoun cannot refer to senators who do not admire Carter, which is the complement set. According to Nouwen (2003), the link to the maximal set can be established only in the case of strong determiner. The anaphoric link to the complement set is rather special. It involves an inference that is generally not licensed, which makes this type of anaphora often marked.

3.4 Inferential property

We have shown the dynamic aspect of quantified expressions as linguistic expressions. In this section, we summarize the logical relations that are derived from the relations between sets.

The quantifiers of type $\langle 1 \rangle$ are classified into two groups: *monotone decreasing* and *monotone increasing*. Each of them is defined as follows, where M is a universe.

Monotone increasing

Q is *monotone increasing* iff for $A \subseteq A' \subseteq M$, $Q(A)$ implies $Q(A')$.

Monotone decreasing

Q is *monotone decreasing* iff for $A' \subseteq A \subseteq M$, $Q(A)$ implies $Q(A')$.

For instance, quantifiers *most* and *no* license the following inferences, where $\Rightarrow$ indicates an entailment relation between natural language sentences.

(3.25) Most students study English hard.
$\Rightarrow$ Most students study English.

(3.26) No student studies English.
$\Rightarrow$ No student studies English hard.

Here, objects satisfying *study English hard* is a subset of objects satisfying *study English*. As mentioned above, a determiner composed with a noun is understood as a type $\langle 1 \rangle$ quantifier. Therefore, *most students* is monotone increasing, while *no student* is monotone decreasing.

Based on the entailment pattern above, the determiner *most* of type $\langle 1, 1 \rangle$ quantifier is said to be *right upward monotone*, and *no*, *right downward monotone*. Here is the definition.

> **Right upward (or downward, resp.) monotonicity**
> Q is right upward (or downward, resp.) monotone iff for $A \subseteq M$ and $B \subseteq B' \subseteq M$, (or $B' \subseteq B \subseteq M$, resp.) $Q(A, B)$ implies $Q(A, B')$.

We often write MON $\uparrow$ and MON $\downarrow$ instead of "right upward monotone" and "right downward monotone," respectively.

Similarly, monotonicity with respect to its left argument can be also defined.

> **Left upward (or downward, resp.) monotonicity**
> Q is left upward (or downward, resp.) monotone iff for $A \subseteq A' \subseteq M$ (or $A' \subseteq A \subseteq M$, resp.) and $B \subseteq M$, $Q(A, B)$ implies $Q(A', B)$.

We often write $\uparrow$MON and $\downarrow$MON instead of "left upward monotone" and "left downward monotone," respectively.

It is also known that all natural language quantifiers are *conservative*. The definition is as follows.

> **Conservativity**
> Q is conservative iff for $A, B \subseteq M$, $Q(A, B) \Leftrightarrow Q(A, A \cap B)$.

In the case of quantifier *most,* for instance, we can see if (3.27a) is true then (3.27b) is also true, and vice versa.

Quantifiers	Conservativity	Monotonicity			
		MON↑	MON↓	↑MON	↓MON
some	✓	✓		✓	
at least n	✓	✓		✓	
every	✓	✓			✓
most	✓	✓			
no	✓		✓		✓
at most n	✓		✓		✓
exactly n	✓				

Table 3.2: Quantifiers and their inferential properties.

(3.27) a. Most swans are white.

b. Most swans are white swans.

Table 3.2 shows basic patterns of inferences associated with quantifiers. There are other inference patterns such as symmetry and reflexivity, which we have omitted from the table.

At first glance, the inferential properties of quantifiers appear to be well-established and straightforward. It has been pointed out, however, that the monotonicity inference in the presence of donkey anaphora poses a puzzle (Benthem, 1987; Kanazawa, 1993, 1994). Consider the contrast between the following two monotonicity inference for the left downward monotone quantifier *no*.

(3.28) No farmer who owns a^1 donkey beats it_1.
⇒ No farmer who owns a^2 female donkey beats it_2.

(3.29) No man who owns a^1 house sprinkles it_1.
⇏ No man who owns a^2 garden sprinkles it_2.

(3.28) follows the inference pattern of left downward monotonicity, since *farmer who owns a female donkey* is a subset of *farmer who owns a donkey*. For (3.29), let us first assume that *man who owns a garden* is a subset of *man who owns a house*, that is, if a man owns a garden, he also owns a house. Although we expect that (3.29) also follows the inference pattern of left downward monotonicity under this assumption, the inference is invalid in this case.

3.5 Previous approaches

GQs are well studied within the model-theoretic approach to natural language semantics. In this approach, the meanings of GQs are defined as relations between sets, as we have explained so far (Barwise and Cooper, 1981). This model-theoretic conception of GQs is also adopted by discourse representation theory (DRT; Kamp and Reyle, 1993; Kamp et al., 2011) and applied to discourse phenomena such as donkey anaphora.

As we have shown, DTS is a proof-theoretic semantics based on dependent type theory. Even if meaning of quantifiers has been well-discussed in the model-theoretic meaning, to give an analysis within a type-theoretic framework is not a trivial task; in particular, it is challenging to give analysis that maintains both logical and dynamic aspects of quantifiers .

In recent years, frameworks based on dependent type theory have been developed as rich proof-theoretic semantics that can handle natural language dynamics, including anaphora and presupposition (Sundholm, 1986, 1989; Ranta, 1994; Pérez, 1995; Krahmer and Piwek, 1999; Mineshima, 2008; Luo, 2012b; Mineshima, 2013; Bekki, 2014; Chatzikyriakidis and Luo, 2014; Bekki and Mineshima, 2017). Although a framework based on dependent

type theory has the potential to deal with both the inferential and dynamic aspects of quantifiers, an analysis of quantifiers has not been pursued much so far.

One of the few is the study by Sundholm (1989). It gives an explicit set-theoretic definitions of quantifiers in terms of logical operators provided in the dependent type theory. The definition captures the basic meaning of quantifiers such as *most*, namely, their truth-conditional meaning that has been associated with the relation between sets. Although the definition also captures the internally dynamic nature of quantifiers, the interaction of quantifiers with anaphora has not been sufficiently explored.

After Sundholm (1986, 1989), Ranta (1994) showed the potential of dependent types with broad empirical data coverage. However, the study provides little analysis of quantifiers in general. In Modern Type Theory (Luo, 2012b; Chatzikyriakidis and Luo, 2014), Lungu and Luo (2014) give an analysis of monotonicity reasoning. While they define rules for monotonicity reasoning of the binary quantifiers, the study does not provide an analysis of the meaning of the sentences containing those quantifiers in general. In particular, the study does not deal with the analysis of sentences containing both quantifiers and anaphora. Although Π-types in dependent type theory can capture the internally dynamic property of *every*, it is not clear how the property can be taken into account for other quantifiers. Chatzikyriakidis and Luo (2018) provides a semantic representation of numeral quantifiers *three*. Luo (2020) proposes a way to introduce a notion of *proof irrelevance* into Martin-Löf type theory and provides an account of the quantifier *most* that can avoid the proportion problem. However, an overall picture of the analysis of anaphora and inference phenomena generally involved in GQs has not been explored.

There are at least two possible approaches to analyze quantifiers in our proof-theoretic setting.

One is what we call *explicit approach*. This is the approach adopted by Sundholm (1989), which is to give an explicit definitions of quantifiers in terms of logical operators provided in the system.

Another approach is *implicit approach*, which is well-studied in proof-theoretic semantics of natural language. This approach is to construct a proof system which contains quantifiers as primitives in the formal system. The study of natural logic (e.g. van Benthem, 2008; Moss, 2010; Icard and Moss, 2014) and the study of natural deduction system containing quantifiers (Francez and Ben-Avi, 2014) are subsumed under this approach. One of the main attractions of such an approach is its computational efficiency. In particular, approaches that employ natural logic and recognize textual entailment by using a logical prover have been successful in computational linguistics (MacCartney and Manning, 2008; Angeli and Manning, 2014; Abzianidze, 2017).

However, at present, the studies adopting implicit approach mainly deal with proof-theoretic inference with quantifiers and they do not take into account dynamics. Thus, it is not clear how to account for the interaction between quantifiers and anaphora, which has been studied in the dynamic semantic frameworks.

Here, we adopt the explicit approach. By giving a semantic representation of quantifiers in an explicit manner, we provide an analysis of quantifier meaning that can account also for their dynamic aspects.

Chapter 4

Quantifier and Anaphora

In this section, we propose a semantic representation of quantifiers in DTS. We first explain Sundholm (1989)'s explicit definition of quantifiers in dependent type theory, which provides the basis of our analysis. In DTS, semantic representation of a sentence represents the meaning of the sentence and also serves as a context for anaphora. From this point of view, we will examine the appropriate semantic representation of quantifiers within the framework.

4.1 Constructive generalized quantifier

4.1.1 Cardinality

Meaning of expressions such as *at least three students* and *fewer than five committees* are captured in terms of the cardinality of a given set, as Table 3.1 has shown. Here, we will explain the way to introduce the notion of a cardinality in our framework, which is mostly based on the study by

$$\frac{}{\mathbf{Nat} : \mathbf{type}}\ \mathbf{Nat}F \qquad \frac{}{0 : \mathbf{Nat}}\ \mathbf{Nat}I_0 \qquad \frac{n : \mathbf{Nat}}{\mathsf{suc}(n) : \mathbf{Nat}}\ \mathbf{Nat}I_s$$

$$\frac{n : \mathbf{Nat} \quad C : \mathbf{Nat} \to \mathsf{s} \quad e : C(0) \quad f : (k : \mathbf{Nat}) \to C(k) \to C(\mathsf{suc}(k))}{\mathsf{natrec}(n, e, f) : C(n)}\ \mathbf{Nat}E$$

Figure 4.1: Inference rules of natural number ($\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$).

Sundholm (1989)[1].

First of all, we introduce natural numbers. Figure 4.1 shows inference rules of **Nat**. According to the introduction rules, a canonical element of type **Nat** is either 0 or $\mathsf{suc}(n)$ for some $n : \mathbf{Nat}$. For simplicity of notation, we often write 1, 2, and 3 instead of $\mathsf{suc}(0)$, $\mathsf{suc}(\mathsf{suc}(0))$, and $\mathsf{suc}(\mathsf{suc}(\mathsf{suc}(0)))$. The behavior of the operator natrec in the elimination rule is as follows: first it executes n, which yields either 0 or $\mathsf{suc}(a)$ for some $a : \mathbf{Nat}$. If it is 0, then it returns e. Otherwise, it returns $f(a)(\mathsf{natrec}(a, e, f))$ and further execute $\mathsf{natrec}(a, e, f)$. Similarly to the first step, it executes a, which is a number one less than the original n .

$$\begin{aligned} \mathsf{natrec}(0, e, f) &\to_\beta e \\ \mathsf{natrec}(\mathsf{suc}(a), e, f) &\to_\beta f(a)(\mathsf{natrec}(a, e, f)), \text{ where } a : \mathbf{Nat} \end{aligned}$$

Following Aczel (1982), we define the k-element type $\mathbf{M}_k$ as follows.

Definition 1 (Finite sequence)**.** For $k \in \mathbf{Nat}$,

$$\mathbf{M}_k \stackrel{def}{\equiv} \mathsf{natrec}(k, \bot, \lambda x \lambda y.(y \uplus \top))$$

Here, natrec first execute k and returns $\bot$ if k is 0. If k is $\mathsf{suc}(a)$ for some a, it

[1] We accommodate parts of the original notation to our notation. For instance, both **set** and **Prop** are written as **type**.

continues to execute $\mathsf{natrec}(a, \bot, \lambda x \lambda y.(y \uplus \top)) \uplus \top$.[2] Thus, $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, ... are given as follows.

$$\begin{aligned}
\mathbf{M}_0 &\equiv \bot \\
\mathbf{M}_1 &\equiv \bot \uplus \top \\
\mathbf{M}_2 &\equiv (\bot \uplus \top) \uplus \top \\
\mathbf{M}_3 &\equiv ((\bot \uplus \top) \uplus \top) \uplus \top \\
&\vdots
\end{aligned}$$

Therefore, $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, ... are types that has zero, one, two, three, ... elements, respectively. $\mathbf{M}_0$ inhabits no term, because it is equivalent to $\bot$. As for $\mathbf{M}_1$, we have $\mathsf{inr}(\langle\rangle)$, where $\langle\rangle : \top$. $\mathbf{M}_2$ has two elements, $\mathsf{inl}(\mathsf{inr}(\langle\rangle))$ and $\mathsf{inr}(\langle\rangle)$, which is verified below.

$$\dfrac{\dfrac{\bot : \mathbf{type} \quad \langle\rangle : \top}{\mathsf{inr}(\langle\rangle) : \bot \uplus \top}\uplus I_R \qquad \top : \mathbf{type}}{\mathsf{inl}(\mathsf{inr}(\langle\rangle)) : (\bot \uplus \top) \uplus \top}\uplus I_L \qquad \dfrac{\begin{array}{c}\vdots \\ \bot \uplus \top : \mathbf{type}\end{array} \quad \langle\rangle : \top}{\mathsf{inr}(\langle\rangle) : (\bot \uplus \top) \uplus \top}\uplus I_R$$

Similarly, $\mathbf{M}_3$ has three elements, $\mathsf{inl}(\mathsf{inl}(\mathsf{inr}(\langle\rangle)))$, $\mathsf{inl}(\mathsf{inr}(\langle\rangle))$, and $\mathsf{inr}(\langle\rangle)$.

Now we have special types that have a specific number of elements, e.g., $\mathbf{M}_3$ having three elements, $\mathbf{M}_5$ having five elements, and so on. We can account for the number of elements of a type by using an injection and a surjection from $\mathbf{M}_k$ to the given type. In dependent type theory, their definitions are given as follows.

Definition 2 (Injection)**.** For any A, B : **type** and $f : A \to B$,

$$\mathbf{injection}\,(f) \stackrel{def}{\equiv} (y : A) \to (y' : A) \to (f\,y =_B f\,y') \to (y =_A y')$$

[2] Aczel (1982) and Sundholm (1989) define the finite sequence with *universe* of small types (Martin-Löf, 1984). For our purpose, we adopt $\mathbf{Nat}E$ that concerns both **type** and kind. It allows us to recursively construct types.

Definition 3 (Surjection)**.** For any A, B : **type** and $f : A \to B$,

$$\textbf{surjection}\,(f) \overset{def}{\equiv} (z : B) \to (y : A) \times (z =_B f\,y)$$

Bijections are defined as follows.

Definition 4 (Bijection)**.** $\textbf{bijection}\,(f) \overset{def}{\equiv} (\textbf{injection}\,(f) \times \textbf{surjection}\,(f))$

Based on these machineries, we can represent various quantifiers. For instance, we can represent the sentence *three farmers are diligent* as follows.[3]

$$\left[\begin{array}{l} f : \mathbf{M}_3 \to \left[\begin{array}{l} x : \textbf{entity} \\ \textbf{farmer}(x) \end{array}\right] \\ \left[\begin{array}{l} \textbf{injection}\,(f) \\ (y : \mathbf{M}_3) \to \textbf{diligent}(\pi_1(f\,y)) \end{array}\right] \end{array}\right] \tag{4.1}$$

What this formula roughly says is as follows: first, there is a mapping f, which is an injection from $\mathbf{M}_3$ to $(x : \textbf{entity}) \times \textbf{farmer}(x)$. Existence of an injection from a three-element set to a set of farmers means that there is (at least) three farmers. Moreover, $(y : \mathbf{M}_3) \to \textbf{diligent}(\pi_1(f\,y))$, where $\pi_1(f\,y)$ refers to an entity that is a farmer, requires that those three entities are all diligent. Note that the functional term f is used to construct a term that fills an argument of the predicate **diligent**. Consequently, when (4.1) is true, there are (at least) three elements x of type **entity** that make both $\textbf{farmer}(x)$ and $\textbf{diligent}(x)$ true.

Along this line, Sundholm (1989) provides a definition of certain types of quantifiers including *finitely many*, *there are more ... than ...*, and *most*.

[3] Here we use *three* as an simple example to show how the finite sequence and injection work for counting. However, it has been argued that numeral expressions such as *three* are not quantifiers themselves (Link, 1987). We will come back to this point later in Section 4.4.

He argues, however, this definition needs to be revised in order to count the farmers correctly. We will come back to this point later.

4.1.2 Notion of *more than half*

Sundholm (1989) interprets *most* as *more than half* (i.e., adopting the definition $|A \cap B| > |A|/2$) and provides its constructive definition. In order to make sense of saying "more than half of A," A should contain a finite number of elements. Thus, we first need the notion of finiteness, which is defined as follows.

Definition 5 (Finiteness)**.** For any A : **type**,

$$\mathbf{Finite}\,(A) \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} f : \mathbf{M}_k \to A \\ \mathbf{bijection}\,(f) \end{array}\right] \end{array}\right]$$

According to the definition, a proof term of **Finite** (A) is a tuple whose first element is some natural number. As f is a bijection, this number corresponds to the exact number of elements in A.

Next, we need a way to obtain the natural number that corresponds to the majority of a given number, e.g., 3 if either 4 or 5 is given. First, the operator **sg** is defined as follows.

Definition 6 (Signum)**.** For any n : **Nat**,

$$\mathbf{sg}(n) \overset{def}{\equiv} \mathsf{natrec}(n, 1, \lambda x \lambda y.0)$$

Here, natrec first executes n. If n is 0, then it returns 1; otherwise, it returns 0. Thus, **sg** behaves as follows.

$$\begin{array}{rcll} \mathbf{sg}(0) & \equiv_\beta & 1 & \\ \mathbf{sg}(\mathsf{suc}(a)) & \equiv_\beta & 0 & where\ a : \mathbf{Nat} \end{array}$$

The operator $\mathbf{rem}_2$, which gives the remainder of a number divided by 2, is then defined as follows by using **sg**.

Definition 7 (Remainder)**.** For n : **Nat**,

$$\mathbf{rem}_2(n) \overset{def}{\equiv} \mathsf{natrec}(n, 0, \lambda x \lambda y.\mathbf{sg}(y))$$

Again, natrec first executes n and returns 0 if n is 0. If n is $\mathsf{suc}(a)$ for some a : **Nat**, it continues to execute $(\lambda x \lambda y.\mathbf{sg}(y))(a)(\mathbf{rem}_2(a)) \equiv \mathbf{sg}(\mathbf{rem}_2(a))$, where $\mathbf{rem}_2(a)$ is recursively computed until the argument reaches to 0.

$$\begin{aligned} \mathbf{rem}_2(0) &\equiv_\beta 0 \\ \mathbf{rem}_2(\mathsf{suc}(a)) &\equiv_\beta \mathbf{sg}(\mathbf{rem}_2(a)) \end{aligned}$$

The operator that gives the quotient of a number divided by 2 is defined as follows with $\mathbf{rem}_2$.

Definition 8 (Integral part)**.** For n : **Nat**,

$$[n/\mathbf{2}] \overset{def}{\equiv} \mathsf{natrec}(n, 0, \lambda x \lambda y.(y + \mathbf{rem}_2(x)))$$

According to the definition, $[\cdot/\mathbf{2}]$ is executed as follows.

$$\begin{aligned} [0/\mathbf{2}] &\equiv_\beta 0 \\ [\mathsf{suc}(a)/\mathbf{2}] &\equiv_\beta [a/\mathbf{2}] + \mathbf{rem}_2(a) \end{aligned}$$

Thus, for $k = 0, 1, 2, ...$, $[k/\mathbf{2}]$ gives each of the following numbers that

corresponds to the integral part of k divided by 2.

$$\begin{aligned}
[0/\mathbf{2}] &= 0 \\
[1/\mathbf{2}] &= [0/\mathbf{2}] + \mathbf{rem}_2(0) = 0 + 0 = 0 \\
[2/\mathbf{2}] &= [1/\mathbf{2}] + \mathbf{rem}_2(1) = 0 + 1 = 1 \\
[3/\mathbf{2}] &= [2/\mathbf{2}] + \mathbf{rem}_2(2) = 1 + 0 = 1 \\
[4/\mathbf{2}] &= [3/\mathbf{2}] + \mathbf{rem}_2(3) = 1 + 1 = 2 \\
[5/\mathbf{2}] &= [4/\mathbf{2}] + \mathbf{rem}_2(4) = 2 + 0 = 2 \\
&\vdots
\end{aligned}$$

We are now able to represent the majority of $n : \mathbf{Nat}$ by $[n/\mathbf{2}] + 1$. Hereafter, we write $\mathbf{majorityOf}(n)$ for $[n/\mathbf{2}] + 1$.

Lastly, the definition of $\geq$ is given as follows.

Definition 9 ($\geq$)**.** For any $m,\ n : \mathbf{Nat}$,

$$m \geq n \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ m =_{\mathbf{Nat}} n + k \end{array} \right]$$

4.1.3 Definition of Most

With the above preparation, Sundholm (1989) defines **Most** for arbitrary $A : \mathbf{type}$, $\phi : A \to \mathbf{type}$, and $a : \mathbf{Finite}(A)$ as follows.

$$\Gamma,\ A : \mathbf{type},\ \phi : A \to \mathbf{type},\ a : \mathbf{Finite}(A) \vdash \mathbf{Most}(A, \phi) : \mathbf{type},$$

$$\text{where } \mathbf{Most}(A, \phi) \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}(\pi_1(a)) \\ \left[\begin{array}{l} f : \mathbf{M}_k \to A \\ \left[\begin{array}{l} \mathbf{injection}(f) \\ (y : \mathbf{M}_k) \to \phi(f\,y) \end{array} \right] \end{array} \right] \end{array} \right] \end{array} \right]$$

This formula represents that more than half of A are ϕ. Here, $a : \mathbf{Finite}(A)$ is imposed as a presupposition in order to make sense of $\mathbf{Most}(A, \phi)$.[4] As the first projection of the term $a : \mathbf{Finite}(A)$, $\pi_1(a)$, is the number of elements in A, k is more than half of A's cardinality. Function f is an injection that maps every element in $\mathbf{M}_k$ to an element in A that satisfies ϕ. When such f exists, more than half of A's elements certainly satisfy ϕ. This captures the intended meaning of *most A are* ϕ.

Note that the Sundholm's definition is based on the treatment of common nouns as types. Hence, ϕ is of type $A \to \mathbf{type}$. Since common nouns are not types but predicates in DTS, we need to modify this definition slightly for our purpose as follows.

$$\Gamma,\ N : \mathbf{entity} \to \mathbf{type},\ V : \mathbf{entity} \to \mathbf{type},\ a : \mathbf{Finite}\left((x : \mathbf{entity}) \times N(x)\right) \vdash \mathbf{Most}(N, V) : \mathbf{type},$$

$$\text{where } \mathbf{Most}(N, V) \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}(\pi_1(a)) \\ \left[\begin{array}{l} f : \mathbf{M}_k \to (x : \mathbf{entity}) \times N(x) \\ \left[\begin{array}{l} \mathbf{injection}(f) \\ (y : \mathbf{M}_k) \to V(\pi_1(fy)) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

Here, both N and V are of type $\mathbf{entity} \to \mathbf{type}$ and the term a is a proof term of $\mathbf{Finite}\left((x : \mathbf{entity}) \times N(x)\right)$.

4.1.4 Proportion problem

Sundholm (1989) pointed out, however, that there is a problem to adopt the above definition: this leads to the well-known *proportion problem* (Kad-

[4] In Sundholm (1989)'s formalism, it is not clear how this presupposition can be handled in the compositional setting. In DTS, such presupposition is represented as a lexical meaning of *most* in terms of @-term, as we will see later.

mon, 1987; Heim, 1990). This problem is not limited to the case of *most* but is mainly caused by the definitions of injection and surjection given earlier.

This becomes clearer if we consider a sentence where the restrictor contains existential quantification of another entity. To simplify the demonstration, we consider the quantifier *three*.

(4.2) Three farmers who own a donkey are diligent.

Here, the restrictor of the quantifier is *farmer who own a donkey*, where we consider the situation that each of the quantified farmers own a donkey. Again, similarly to (4.1), the semantic representation of (4.2) can be given as follows with an injection.

$$
(4.3)\quad \left[\begin{array}{l} f : \mathbf{M}_3 \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} z : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(z) \\ \mathbf{own}(x,z) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} \mathbf{injection}\,(f) \\ (y : \mathbf{M}_3) \to \mathbf{diligent}(\pi_1(f\,y)) \end{array}\right] \end{array}\right]
$$

Here, f should be a function from $\mathbf{M}_3$ to a Σ-type that consists of nested pairs: each term of the Σ-type would be a 5-tuple $\langle e_1, p_{\mathbf{f}(e_1)}, e_2, p_{\mathbf{d}(e_2)}, p_{\mathbf{o}(e_1,e_2)}\rangle$, where each of the participated terms is understood as follows.[5]

[5] We often abbreviate nested pair $\langle x_1, \langle x_2, x_3\rangle\rangle$ as $\langle x_1, x_2, x_3\rangle$.

e_1	an entity who is a farmer
$p_{\mathbf{f}(e_1)}$	a proof that the entity e_1 is a farmer
e_2	an entity which is a donkey
$p_{\mathbf{d}(e_2)}$	a proof that the entity e_2 is a donkey
$p_{\mathbf{o}(e_1,e_2)}$	a proof that e_1 owns e_2

Suppose that there is only one farmer, f_a, who owns a donkey. He owns three donkeys d_{a1}, d_{a2}, and d_{a3}. We also have a proof that f_a is a diligent person. In this situation, we have the following three 5-tuples.

(4.4) $\langle f_a, p_{\mathbf{f}(f_a)}, d_{a1}, p_{\mathbf{d}(d_{a1})}, p_{\mathbf{o}(f_a,d_{a1})}\rangle$
$\langle f_a, p_{\mathbf{f}(f_a)}, d_{a2}, p_{\mathbf{d}(d_{a2})}, p_{\mathbf{o}(f_a,d_{a2})}\rangle$
$\langle f_a, p_{\mathbf{f}(f_a)}, d_{a3}, p_{\mathbf{d}(d_{a3})}, p_{\mathbf{o}(f_a,d_{a3})}\rangle$

According to the definition of **injection** (Definition 2), an injection from $\mathbf{M}_3$ exists because we have three tuples. As we have assumed that the first component of each tuple, f_a, is diligent, the semantic representation is true even if there is only one farmer, which is counter-intuitive. Here we see a typical situation of the proportion problem that is caused by quantifying over tuples.

In the standard setting, the proportion problem is associated with a sentence where the quantifier restrictor contains the relative clause with an indefinite in it. In the proof-theoretic setting, however, quantifying over tuples may cause problem even in absence of indefinite.

Consider the simpler variant of (4.2).

(4.5) Three farmers are diligent.

Now, similarly to (4.3), there exists a function from $\mathbf{M}_3$ to the Σ-type $(x : \mathbf{entity})\times$ $\mathbf{farmer}(x)$. This Σ-type consists of pairs $\langle e_1, p_{\mathbf{f}(e_1)}\rangle$, where $p_{\mathbf{f}(e_1)}$ is again a proof of $\mathbf{farmer}(e_1)$. This means that not only entities but also proposi-

tions such as **farmer**(e_1) introduce a proof term. Quantifying over tuples (pairs) may then cause problem because one proposition can have multiple different proofs.

Consider the proposition "John is diligent" and let us assume that John is a Canadian policeman. If we have the proof term p_1 of that assumption , as well as the proof term p_2 that every Canadian is diligent, we can construct a proof that John is diligent by combining p_1 and p_2.

$$john : \mathbf{entity},\ p_1 : \begin{bmatrix} \mathbf{Canadian}(john) \\ \mathbf{policeman}(john) \end{bmatrix},\ p_2 : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{Canadian}(x) \end{bmatrix} \right) \to \mathbf{diligent}(\pi_1 u)$$
$$\vdash\ p_2(john, \pi_1 p_1) : \mathbf{diligent}(john)$$

Now, suppose that we have another proof term p_3 that every policeman is diligent. We can construct a proof that John is diligent also by combining p_1 and p_3.

$$john : \mathbf{entity},\ p_1 : \begin{bmatrix} \mathbf{Canadian}(john) \\ \mathbf{policeman}(john) \end{bmatrix},\ p_3 : \left(u : \begin{bmatrix} x : \mathbf{entity} \\ \mathbf{policeman}(x) \end{bmatrix} \right) \to \mathbf{diligent}(\pi_1 u)$$
$$\vdash\ p_3(john, \pi_2 p_1) : \mathbf{diligent}(john)$$

In this way, we can obtain two different proof terms for the proposition "John is diligent." Therefore, counting pairs may cause the problem even in the absence of an indefinite.

To avoid this problem, Luo (2020) proposes to extend Martin-Löf type theory with the logic of *mere propositions* called h-logic and introduce the notion of *proof irrelevance*. Proof irrelevance is a principle that any two proofs of the same logical proposition should be the same. This can solve the multiple proofs problem noted above, but is usually not available for predicative type theories such as Martin-Löf type theory, where every proposition is also a type: for instance, two entities a and b of type

entity are identified, which is an unwanted result. Since the extended system have mere propositions which are distinguished from type, it can have a rule that every two proofs of a mere proposition are equal.

Luo (2020) suggests that one can also avoid the proportion problem by defining a weak existential quantifier $\exists$ in terms of mere propositions and use it to represent the relative clause of the donkey sentence instead of Σ. The resultant semantic representation, however, seems to block some sort of inter-sentential anaphoric link (see, Section 5.6).

To maintain the anaphoric accessibility, we will crucially use Σ-type also for the relative clause of the donkey sentence. In order to avoid counting same entities multiple times as in (4.4), we follow Sundholm's proposal and use different version of injection, surjection, and bijection, which identify tuples by its first component, namely, an object of type **entity** in our setting. Note that **entity** in DTS can be defined as an enumeration type. For now, we assume that for any objects of **entity**, we know its canonical form and thus can distinguish one from the other.

4.1.5 Counting entities

Sundholm (1989) defines the following injection where two tuples are identified regarding to their first component.[6]

Definition 10 (**injection**$_{\pi_1}$)**.** For any A, B : **type**, $C : B \to$ **type**, and $f : A \to (x : B) \times C$,

$$\textbf{injection}_{\pi_1}(f) \overset{def}{\equiv} (y : A) \to (y' : A) \to \pi_1(fy) =_B \pi_1(fy') \to y =_A y'$$

[6] Since Sundholm's original definition contains a type mismatch, we present a slightly modified version of it here.

Here, f is a function whose range is a Σ-type, and $\mathbf{injection}_{\pi_1}(f)$ says that f is an injection which is relativized to the first component B. Two terms y and y' of A are equated if the first components of $f(y)$ and $f(y')$ are the same, regardless of the rest of the tuple. The following theorem immediately follows by definition.

Theorem 1: injection and injection$_{\pi_1}$

For any A, B : **type**, $C : B \to$ **type**, and $f : A \to (x : B) \times C$, if $\mathbf{injection}_{\pi_1}(f)$, then $\mathbf{injection}\,(\lambda x.\pi_1(f(x)))$.

Proof. Assume $\mathbf{injection}_{\pi_1}(f)$. Then the following holds by definition.

$$(y, y' : A) \to \pi_1(fy) =_B \pi_1(fy') \to y =_A y'$$

Assume we have $y : A$, $y : A'$, and $(\lambda x.\pi_1(f(x)))y =_B (\lambda x.\pi_1(f(x)))y'$. From $(\lambda x.\pi_1(f(x)))y \to_\beta \pi_1(f(y))$ and $(\lambda x.\pi_1(f(x)))y' \to_\beta \pi_1(f(y'))$, we have

$$\pi_1(f(y)) =_B \pi_1(f(y')).$$

Thus, by $\mathbf{injection}_{\pi_1}(f)$, we obtain $y =_A y'$. By Π-introduction, we get

$$(y, y' : A) \to (\lambda x.\pi_1(f(x)))y =_B (\lambda x.\pi_1(f(x)))y' \to y =_A y'$$

End of proof.

Surjection is defined analogously.

Definition 11 ($\mathbf{surjection}_{\pi_1}$.)**.** For any A, B : **type**, $C : B \to$ **type**, and $f : A \to (x : B) \times C$,

$$\mathbf{surjection}_{\pi_1}(f) \stackrel{def}{\equiv} (z : (x : B) \times C) \to \begin{bmatrix} y : A \\ \pi_1(z) =_B \pi_1(fy) \end{bmatrix}$$

While the standard surjection requires that there exists a corresponding element of A for every tuple z, $\mathbf{surjection}_{\pi_1}$ only requires that the corresponding element exists for z's first component. For instance, if there are $\langle b, c_1\rangle$ and $\langle b, c_2\rangle$, it is enough to have only one y such that $fy =_{(x:B)\times C} \langle b, c_1\rangle$ because $\pi_1\langle b, c_2\rangle =_B \pi_1(fy) =_B b$.

Bijection is also defined in the same way as before.

Definition 12 ($\mathbf{bijection}_{\pi_1}$)**.** For any A, B : **type**, $C : B \to$ **type**, and $f : A \to (x : B) \times C$,

$$\mathbf{bijection}_{\pi_1}(f) \overset{def}{\equiv} \Big(\mathbf{injection}_{\pi_1}(f) \times \mathbf{surjection}_{\pi_1}(f)\Big)$$

Accordingly, $\mathbf{finite}_{\pi_1}$, which adopts $\mathbf{bijection}_{\pi_1}$, is defined as follows.

Definition 13 (Finiteness)**.** For any B : **type** and $C : B \to$ **type**,

$$\mathbf{finite}_{\pi_1}(x : B) \times C \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} f : \mathbf{M}_k \to (x : B) \times C \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right]$$

Finally, we can revise the definition of **Most** as follows, where $\mathbf{finite}_{\pi_1}$ and $\mathbf{injection}_{\pi_1}$ are abbreviated notations of $\mathbf{finite}_{\pi_1}$ and $\mathbf{injection}_{\pi_1}$, respectively.

(4.6)

$$\Gamma, N : \mathbf{entity} \to \mathbf{type}, V : \mathbf{entity} \to \mathbf{type}, a : \mathbf{finite}_{\pi_1}\big(x : \mathbf{entity}\big) \times N(x) \vdash \mathbf{Most}(N, V) : \mathbf{type},$$

where $\mathbf{Most}(N, V) \overset{def}{\equiv}$

$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}(\pi_1(a)) \\ \left[\begin{array}{l} f : \mathbf{M}_k \to \big(x : \mathbf{entity}\big) \times N(x) \\ \left[\begin{array}{l} \mathbf{injection}_{\pi_1}(f) \\ (y : \mathbf{M}_k) \to V(\pi_1(fy)) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

The range of f corresponds to the quantified objects, which is given by the restrictor noun phrase of *most*. In DTS, such objects are always represented by a Σ-type of $(x : \textbf{entity}) \times N(x)$ for some predicate N, whose first component is type **entity**. Thus, we can quantify over entities instead of tuples.

Here is a remark on the uniformity problem pointed out by Sundholm (1989). Since Sundholm (1989) treats common nouns as types, the range of f can be any type, but $\textbf{injection}_{\pi_1}$ is applicable only when it is a Σ-type. Thus, as Sundholm himself pointed out, one needs to define **Most** by case analysis on A.

4.2 Analysis of *most* in DTS

Sundholm (1989) sketched the constructive definition of **Most** in a way that can avoid the so-called proportion problem. Although the study provides an essential tool-set to translate the traditional set-theoretic quantifier analysis into Martin-Löf type theory, an analysis of the natural language quantifier is not yet well explored. In particular, as we have pointed out previously, an analysis that can deal with both dynamic linguistic phenomena and the proof-theoretic inference of quantifiers has not been well studied. In this section, we will provide the semantic representation of the quantifier *most* in DTS on the basis of the formalism by Sundholm (1989).

Given the linguistic behavior of the quantifier *most*, we can say that there are at least two points to be improved in the definition (4.6).

One is the *existential presupposition*. Quantifier *most* is known as a *strong determiner*: the sentence *Most farmers own a donkey* presupposes that there exists a farmer, not only presupposing the number of farmers

is finite.

Another is the *maximization* of the number. When **Most**(N,V) is true (i.e., there exists a proof term of **Most**(N,V)), we will have a natural number k and a function f of $\mathbf{M}_k \rightarrow (x : \mathbf{entity}) \times N(x)$ as part of the proof. However, k is not necessarily the number of entities in question: all we can say is that it is a number greater than the majority of entities in $(x : \mathbf{entity}) \times N(x)$. As a proof term can behave as a discourse referent, we think it is crucial that k is the exact number of entities.

Besides, we need to devise a way to give semantic representations for other quantifiers in the same way. This is because the definition employs $\mathbf{injection}_{\pi_1}$. Due to its nature, the existence of an injection from $\mathbf{M}_k$ to the target entities ensures that there are *at least* k entities. On the contrary, it cannot guarantee that there are *exactly* k entities or *at most* k entities, which is necessary to represent the meaning of quantifiers such as *exactly three* and *at most three*.

Therefore, we define **Most** as follows.

(4.7) **Most** in DTS

For $\Gamma \vdash N : \mathbf{entity} \rightarrow \mathbf{type}$ and $\Gamma,\ x : \mathbf{entity},\ u : N(x) \vdash V : \mathbf{entity} \rightarrow \mathbf{type}$, $\Gamma \vdash \mathbf{Most}(N,V) : \mathbf{type}$,

where $\mathbf{Most}(N,V) \overset{def}{\equiv}$

$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}\left(\pi_1\pi_1\left(@ :: \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ N(x) \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right]\right)\right) \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

Lexical entry of *most* is then given as follows.

$$\text{most};\ (S/(S\backslash NP))/N;\ \lambda n\lambda v.\mathbf{Most}(n,v)$$

Here we represent the presupposition of *most* by the underspecified term in DTS. The annotated type of @-term expresses the presupposition. This captures that the quantifier *most* triggers the presupposition that the quantification domain is finite and non-empty.

The resolution of @-term proceeds along the same line as the case of existential presupposition of definite descriptions (see Section 2.6.1). By the type checking and proof search in DTS, it will be eventually replaced with a concrete proof term $\langle a, e\rangle$ where a is a proof of finiteness and e is a proof that the number of elements is greater than or equal to 1. From the first component, a, we can obtain its first component $\pi_1(a)$ which is the number of entities in $(x : \mathbf{entity}) \times N(x)$ (Definition 13). On the other hand, the second component e is discarded: it is only used to specify that $\pi_1(a) \geq 1$, which is needed to presuppose the existence of an element.

To maximize the natural number k, we use $\mathbf{bijection}_{\pi_1}$, which is the conjunction of $\mathbf{injection}_{\pi_1}$ and $\mathbf{surjection}_{\pi_1}$ instead of only $\mathbf{injection}_{\pi_1}$. This is because $\mathbf{surjection}_{\pi_1}(f)$ ensures that every entities in $(x : \mathbf{entity}) \times (u : N(x)) \times V(x)$ has a corresponding element in $\mathbf{M}_k$. The subsequent discourse may use this number k in the proof search for @-term resolution (see Section 4.5). Also, this enables us to account for quantifiers such as *exactly three* as well as monotone decreasing cases such as *at most three*.

As we mentioned in Section 3.1, the quantifier *most* can be defined also as $|A \cap B| > |A - B|$. If we adopt this definition, the semantic representation of *most* can be given as follows.

$$\textbf{Most}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} u : \left[\begin{array}{l} k : \textbf{Nat} \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u' : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \textbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} v : \left[\begin{array}{l} k' : \textbf{Nat} \\ \left[\begin{array}{l} f' : \mathbf{M}_{k'} \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} v' : N(x) \\ \neg V(x) \end{array}\right] \end{array}\right] \\ \textbf{bijection}_{\pi_1}(f') \end{array}\right] \end{array}\right] \\ \pi_1 u > \pi_1 v \end{array}\right] \end{array}\right] \tag{4.8}$$

Here, $\pi_1 u > \pi_1 v$ compares the cardinality of $\big(x : \textbf{entity}\big) \times (u : N(x)) \times V(x)$ and $\big(x : \textbf{entity}\big) \times (u : N(x)) \times \neg V(x)$.

If these two semantic representations at least represent the same cardinal condition, how do we choose one over the other? One possiblity is that, as (Hackl, 2009) argued, to choose the second one based on the difference in their verification procedure. For this particular formulation, however, we prefer the first version over the second one from the view point of its anaphoric potential. Since semantic representation (4.8) has a parallel structure, it is natural to expect that u and v have the same anaphora accessibility. However, since u and v correspond to the reference set and the complement set, respectively, this contrary to the Nouwen (2003)'s observation, where the reference to complement set is generally not available.

Although it may be still possible to explain the difference in anaphora accessibility by assuming other factors, here we adopt the first one given above as the semantic representation of *most* in our analysis.

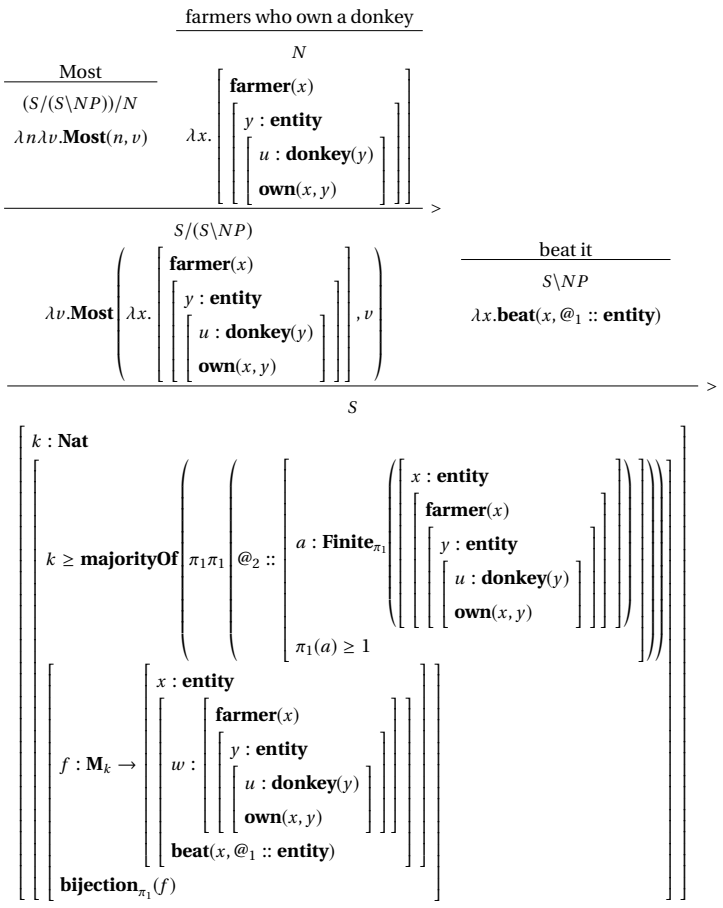


Figure 4.2: Semantic composition of (4.9).

4.3 Donkey sentence

The semantic representation of *most* captures the internally-dynamic nature of the quantifier. Consider the following donkey sentence.

(4.9) Most farmers who own a[1] donkey beat it$_1$.

The semantic composition is analogous to the universal donkey sentence (2.20) in Section 2.5.1. Hence, the semantic representation of (4.9) is derived as in Figure 4.2. Here, $@_1$ represents the pronoun *it* in the nuclear scope and $@_2$ represents the existential presupposition of *most*.

As the obtained semantic representation should be of sort **type**, the following judgment should hold for $@_2$.

$$\mathcal{K},\ k : \mathbf{Nat} \vdash \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right] \quad \textit{true} \tag{4.10}$$

Let us assume that we have a knowledge that *exactly five farmers own a donkey*. In this case, the global context $\mathcal{K}$ contains the following proof term in (4.11).

$$\langle 5, \mathcal{F}, \mathcal{B} \rangle : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \tag{4.11}$$

Recall that the proof of finiteness consists of a natural number, a function, and a proof that the function satisfies $\mathbf{bijection}_{\pi_1}$, which correspond to 5, $\mathcal{F}$, and $\mathcal{B}$, respectively.

Since there is no preceding discourse in (4.9), there is no option for presupposition filtration for the presupposition represented by $@_2$. Thus, it is resolved by the information in $\mathcal{K}$. As we have (4.11) in $\mathcal{K}$ and can also prove $5 \geq 1$, which we write $\mathcal{P}$, we can construct $\langle\langle 5, \mathcal{F}, \mathcal{B}\rangle, \mathcal{P}\rangle$ that satisfies the judgement (4.10). This proof can substitute for $@_2$.

For $@_1$, the following two judgement should hold.

(4.12)

$$\mathcal{K},\ k : \mathbf{Nat},\ \langle\langle 5, \mathcal{F}, \mathcal{B}\rangle, \mathcal{P}\rangle : \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right], p : k \geq \mathbf{majorityOf}(5),$$

$$z : \mathbf{M}_k,\ x : \mathbf{entity},\ w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \vdash @_1 :: \mathbf{entity} : \mathbf{entity}$$

(4.13)

$$\mathcal{K},\ k : \mathbf{Nat},\ \langle\langle 5, \mathcal{F}, \mathcal{B}\rangle, \mathcal{P}\rangle : \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right], p : k \geq \mathbf{majorityOf}(5),$$

$$f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{beat}(x, @_1 :: \mathbf{entity}) \end{array}\right] \end{array}\right], i : \mathbf{injection}_{\pi_1}(f),\ x : \mathbf{entity},$$

$$w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \vdash @_1 :: \mathbf{entity} : \mathbf{entity}$$

Here, (4.13) is the @-term which appears in $\mathbf{bijection}_{\pi_1}$, in the antecedent clause of $\mathbf{surjection}_{\pi_1}$. This means that, in $\mathbf{Most}(N, V)$, N appears three times and V appears twice in total. If N and V contains @-term, it is duplicated in the semantic representation. In this case, we have to resolve @-term with the same index in a consistent way.

In general, if we have $\Gamma \vdash @_i :: \Lambda : \Lambda$ and $\Gamma' \vdash @_i :: \Lambda : \Lambda$, the proof

term constructed for $@_i :: \Lambda$ under Γ should be also constructed under the context $[\Gamma']$, where $[\Gamma']$ is obtained from Γ' by only renaming the variables.[7] In the current case, we can find $\pi_1\pi_2 w$ that can replace $@_1$, which corresponds to the intended reading of (4.9).

The fully resolved semantic representation is as follows.

(4.14)
$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}(5) \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{beat}(x, \pi_1\pi_2 w) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

[7] The treatment of @-terms presented here is not compatible with the assumption made by Kubota et al. (2019) in their analysis on the interpretive parallelism within DTS.

(i) Every English man thinks, and every American believes, that he is a genius.

In order to obtain the bound reading of the pronoun *he* in the above sentence, they made two assumptions which play key roles in their analysis (quoted from page 4).

a. **Ban on the duplication of underspecified terms:** In a well-formed semantic representation of DTS, an underspecified @-term with the same index can appear at most once.

b. **Normal form requirement on compositionally derived semantic terms:** At each step of semantic composition, the semantic term assigned to the derived linguistic expression is in β-normal form.

Since the restriction (a) invokes anaphora resolution before the β-reduction of the semantic representation takes place, the bound reading can be naturally derived.

In our case, @-term that appears in the object position of V in $\mathbf{Most}(N, V)$ can find its antecedent only after the semantic composition completed. Thus, if we adopt the restriction (a), the semantic composition just fails when **Most** takes V.

As **majorityOf**(5) is 3, the semantic representation can be immediately understood as *at least three farmers who own a donkey beat his donkey*.

Note that the semantic representation in 4.14 corresponds to the existential reading of the donkey sentence (4.9): we count farmers if there exists a proof of an associated donkey in a beating relation. Suppose that there are five farmers and they own at least one donkey and beat some of them. The situation is illustrated below.[8]

farmer	f_a	f_b	f_c	f_d	f_e
donkey	d_{a1}	$d_{b1},\ d_{b2}$	$d_{c1},\ d_{c2}$	d_{d1}	d_{e1}
beat	(f_a, d_{a1})	(f_b, d_{b1})	$(f_c, d_{c1}),\ (f_c, d_{c2})$	-	-
not-beat	-	(f_b, d_{b2})	-	(f_d, d_{d1})	(f_e, d_{e1})

Now, the donkey sentence (4.9) is intuitively true in the existential reading, because three in five farmers, f_a, f_b, and f_c, fulfill the condition. Obviously, (f_a, d_{a1}) can be a destination of the function f. For f_b, (f_b, d_{b1}) can be the destination as well only because it fulfills the condition: the existence of (f_b, d_{b2}), which is not in a beating relation, does not matter. For f_c, either (f_c, d_{c1}) or (f_c, d_{c2}) can be a destination of f. Recall that only one of them can be a destination of f as f meets $\mathbf{injection}_{\pi_1}(f)$. Also, none of f_a, f_b, and f_c is overlooked as f meets $\mathbf{surjection}_{\pi_1}(f)$.

4.4 Other quantifiers

The proposed definition can be naturally extended to other quantifiers such as *at least three, exactly three*, and *at most three*.

We treat *three* in *at least three* as an expression provides numerical

[8] As the range of f is complex Σ-type, the precise elements in the range of f are 6-tuples consists of an entity that is a farmer, a proof of the entity being a farmer, and so on.

specification, whose syntactic category is specified as NUM, and *at least* as an phrasal expression which modifies such a numeral. Their lexical entries are given as follows.[9]

$$\text{three}; NUM; 3$$

$$\text{at least}; S/(S\backslash NP)/N/NUM; \lambda k'.\lambda N.\lambda V.\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq k' \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : Nx \\ Vx \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

Then the semantic representation of quantifier *at least three* is given as follows.

$$\lambda N.\lambda V.\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq 3 \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : Nx \\ Vx \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right] \tag{4.15}$$

Let us abbreviate it as **AtLeastThree**. By similar procedure, we obtain **AtLeastN** in general for the semantic representation of *at least n*, where n corresponds to the natural number $\mathcal{N}$.

Exactly n and *at most n* can be represented in the same way.

[9] Here we simply treat the semantic type of NUM is **Nat**. It is also possible to treat it as an object of type **entity** and define an operation to convert it to an object of type **Nat**.

$$(4.16)\quad \textbf{ExactlyN}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} k : \textbf{Nat} \\ \left[\begin{array}{l} k =_{\textbf{Nat}} \mathcal{N} \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \textbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

$$(4.17)\quad \textbf{AtMostN}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} k : \textbf{Nat} \\ \left[\begin{array}{l} k \leq \mathcal{N} \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \textbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

They are all defined in terms of $\textbf{bijection}_{\pi_1}$. In this way, we can give the semantic representation of the downward monotone quantifier such as *at most n*.

For the bare numeral *three* in the sentence *three farmers are diligent*, there is no expression that takes the numeral and behaves as a quantifier. Here we assume that we have the following empty expression.[10]

[10] Here we treat numerals as themselves are not quantifiers. This view point agrees with studies where numerals are accounted for as expressions modifying nouns (Link, 1987; Krifka, 1996, among others).

Under the numeral-as-modifier approach, expressions such as *the three students* can be naturally handled. Link (1987), however, allows postulating an extra node for numerals to avoid strings like *pretty three girls*. Also, Krifka (1996) assumes a morphologically empty determiner for his analysis of NPs like *two students* in order to derive the quantificational meaning.

Although we adopt different analysis here, the use of NUM and $\varnothing_Q$ is similarly motivated. Based on our current analysis, definite article *the* in *the three N V* can be analyzed as an expression whose syntactic category is $S/(S\backslash NP)/N/NUM$, similarly to *at*

$$\varnothing_Q;\ S/(S\backslash NP)/N/NUM;\ \lambda k.\lambda N.\lambda V.\left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n = k \\ \left[\begin{array}{l} f : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : Nx \\ Vx \end{array}\right] \end{array}\right] \\ \mathbf{injection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

With this bare expression, we can have a quantifier **N** for the bare numeral n in general.

$$\text{(4.18)}\quad \mathbf{N}(N,V) \stackrel{def}{\equiv} \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n =_{\mathbf{Nat}} \mathcal{N} \\ \left[\begin{array}{l} f : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \mathbf{injection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

Note that this semantic representation corresponds to the distributive reading of the numeral expressions, as it is understood as *there are n entities which satisfy both properties N and V*. On the other hand, in the following case, this reading seems to be not the preferred one.

(4.19) Three men lifted the piano.

(4.20) Three men met in the park.

On the distributive reading of (4.19), the sentence means that there were three men each lifting a piano. However, there exists another reading, which seems to be more natural, that there has been at least one joint lifting with three people involved (Link, 1987). The second reading is called collective reading. In (4.20) where the collective verb *met* appears, only

least, which is the different lexical entry we presented in the earlier chapter.

the collective reading seems to be plausible. To account for both the collective reading and the distributive reading in a compositional way, we can integrate the existing analysis of plural objects into our framework. See Link (1983) for the standard approach. Proposals on the treatment of plural objects in a dependently-typed setting has been also made by Boldini (2001) and Chatzikyriakidis and Luo (2013). Since the full discussion of this phenomenon is beyond the scope of this thesis, we will focus on the distributive reading of numeral expressions.

The difference between **ExactlyN** and **N** is that only the former maximize the natural number k in the representation by $\mathbf{bijection}_{\pi_1}$. This ensures that *Exactly n N V* becomes false in the case where there are more than $\mathcal{N}$ elements.

The sentences *At least n N V* and *n N V* have the same truth condition, but we define them in a different way. This is because the determiners behave differently when the noun phrases are referred to from the subsequent sentences. As Kadmon (1987) observed in the following sentences, *ten N* differs from *at least ten N* in anaphora possibilities.

(4.21) [At least ten kids]$_i$ walked into the room. They$_i$ were making an awful lot of noise.

(4.22) [Ten kids]$_i$ walked into the room. They$_i$ were making an awful lot of noise.

Both of the first sentences of (4.21) and (4.22) can be true in the situation where there are actually more than ten kids. Suppose that there are twelve kids who walked into the room. In (4.21), *they* in the second sentence refers to all the twelve kids who walked into the room. By contrast, *they* in (4.22) refers to a collection of ten kids in question. Thus, *at least ten kids* behaves as the maximal collection of kids, while *ten* does not.

We define **AtLeastN** and **N** to reflect this difference in anaphora possibilities. We define **AtLeastN** with the condition $k \geq \mathcal{N}$, where k is maximized by $\mathbf{bijection}_{\pi_1}$. On the other hand, we define **N** with the cardinality condition $k = \mathcal{N}$. It can still be true when there are more than three objects, as we use $\mathbf{injection}_{\pi_1}$.

Quantifiers *some*, *no*, and *every* can be represented in the same manner. Existential reading of *some* and *no* can be represented as follows.

$$\mathbf{Some}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} f : \mathbf{M}_1 \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \mathbf{injection}_{\pi_1}(f) \end{array}\right] \tag{4.23}$$

$$\mathbf{No}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} f : \mathbf{M}_0 \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \tag{4.24}$$

Recall that $\mathbf{M}_1 \equiv \bot \uplus \top$. Then it follows that (4.23) and $(x : \mathbf{entity}) \times (u : N(x)) \times V(x)$ are mutually deducible. Similarly, by $\mathbf{M}_0 \equiv \bot$, (4.24) and $\neg\,(x : \mathbf{entity}) \times (u : N(x)) \times V(x)$ are mutually deducible.

As for *every*, we can have the following semantic representation, which captures the cardinality condition of the restrictor and the nuclear scope.

$$\textbf{Every}(N,V) \overset{def}{\equiv} \left[\begin{array}{l} a : \left[\begin{array}{l} n : \textbf{Nat} \\ \left[\begin{array}{l} f : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ N(x) \end{array}\right] \\ \textbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} \left[\begin{array}{l} g : \mathbf{M}_{\pi_1 a} \rightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \\ \textbf{bijection}_{\pi_1}(g) \end{array}\right] \\ \mathcal{PR}\,(\pi_1 a \geq 1) \end{array}\right] \end{array}\right] \tag{4.25}$$

Here, $\mathcal{PR}\,(A) \overset{def}{\equiv} (@_i :: A) =_A (@_i :: A)$. $\mathcal{PR}$ is the operator which we call *presupposition operator*. It can be used to represent the presupposition that is independent of the assertive content. Since it is defined in terms of an identity, it does not contribute to the assertive content of the semantic representation. In the type checking stage, however, the @-term in the $\mathcal{PR}$ operator triggers the proof search just like other @-terms, which requires that the proposition A is provable under the context. In the current case, the $\mathcal{PR}$ operator triggers the proof search of $\pi_1 a \geq 1$, which requires that the cardinality of the restrictor is at least one. This captures the existential presupposition of the strong determiner *every*.

One crucial problem of (4.25) is that a formula we can deduce from it is weaker than we expect. As both f and g are bijection, we can prove that the following proposition holds.

$$\left[\begin{array}{l} x : \textbf{entity} \\ N(x) \end{array}\right] \leftrightarrow \left[\begin{array}{l} x : \textbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ V(x) \end{array}\right] \end{array}\right] \tag{4.26}$$

This is, however, not sufficient for the meaning of *every*, because the antecedent and the consequence clauses consider different entities. This

motivates us to more directly represent the relation $A \subseteq B$ (not $|A| = |A \cap B|$). More appropriate representation for *every* is as follows.

$$
(4.27)\quad
\left[\begin{array}{l}
u : \left[\begin{array}{l}
a : \left[\begin{array}{l}
k : \mathbf{Nat} \\
\left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ N(x) \end{array}\right] \end{array}\right] \\
\mathbf{bijection}_{\pi_1}(f)
\end{array}\right] \\
\left(z : \mathbf{M}_{\pi_1 a}\right) \rightarrow V(\pi_1 f(z))
\end{array}\right] \\
\mathcal{PR}\left(\pi_1 \pi_1 u \geq 1\right)
\end{array}\right]
$$

Now, we treat that the finiteness of the restrictor is also a part of the presupposition of *every* and represent the inclusive relation between the restrictor and the nuclear scope. The semantic representation in (4.27) corresponds to existential reading. To account for both the universal reading and the existential reading, we have the following semantic representation.

$$
(4.28)\quad
\left[\begin{array}{l}
u : \left[\begin{array}{l}
a : \left[\begin{array}{l}
k : \mathbf{Nat} \\
\left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ N(x) \end{array}\right] \end{array}\right] \\
\mathbf{bijection}_{\pi_1}(f)
\end{array}\right] \\
\left(z : \mathbf{M}_{\pi_1 a}\right) \rightarrow (v : N(\pi_1 f(z))) \rightarrow V(\pi_1 f(z))
\end{array}\right] \\
\mathcal{PR}\left(\pi_1 \pi_1 u \geq 1\right)
\end{array}\right]
$$

In the semantic representation of donkey sentence, e.g., *every farmer who owns a donkey beats it*,he @-term corresponding to the donkey pronoun appears in V. Thus, it can always find at least two antecedent: one is in v and the other is in $\pi_2 f(z)$. If the @-term is resolved by using v, the resultant semantic representation corresponds to the universal reading. If the @-term is resolved by using $\pi_2 f(z)$, it corresponds to the existential reading.

Let us illustrate this by the example of universal donkey sentence. For

simplicity, let us ignore the second part, $\mathcal{PR}(\pi_1\pi_1 u \geq 1)$. The semantic representation of *Every farmer who owns a donkey beats it* is given as follows.

$$
\left[\begin{array}{l}
k : \mathbf{Nat} \\
\left[\begin{array}{l}
f : \mathbf{M}_k \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\
\left[\begin{array}{l} b : \mathbf{bijection}_{\pi_1}(f) \\ (z : \mathbf{M}_k) \rightarrow \left(\begin{array}{l} v : \left[\begin{array}{l} \mathbf{farmer}(\pi_1 f(z)) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(\pi_1 f(z), y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right) \rightarrow \mathbf{beat}(\pi_1 f(z), @_1 :: \mathbf{entity}) \end{array}\right]
\end{array}\right]
\end{array}\right]
\tag{4.29}
$$

The possible resolution for $@_1$ in question is $\pi_1\pi_2 v$ and $\pi_1\pi_2\pi_2(f(z))$.

Let us deduce simpler formula from (4.29). The key is that we have the proof of $\mathbf{bijection}_{\pi_1}(f)$, from which, we can obtain the proof of $\mathbf{surjection}_{\pi_1}(f)$, which we write s_f.

$$
s_f : \left(\begin{array}{l} w : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right) \rightarrow \left[\begin{array}{l} y : \mathbf{M}_k \\ \pi_1 z = \pi_1 f(y) \end{array}\right]
\tag{4.30}
$$

Here, we informally refers to k and f appears in the original formula (4.29), but of course both should be extracted from the proof of (4.29) by Σ-elimination. We also have the following formula, which is the last component of (4.29). ($\mathcal{A}$ represents the proof term which has replaced $@_1$. In the current case, it is either $\pi_1\pi_2 v$ or $\pi_1\pi_2\pi_2(f(z))$.)

$$(4.31)\quad p : (z : \mathbf{M}_k) \to \left(v : \left[\begin{array}{l} \mathbf{farmer}(\pi_1 f(z)) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(\pi_1 f(z), y) \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \mathbf{beat}(\pi_1 f(z), \mathcal{A})$$

(4.30) plays two crucial role. One is, by taking the proof of *farmer who owns a donkey*, returns a proof of $\mathbf{M}_k$, which can be fed to (4.31). The other is to provide the identity relation of the entity.

From (4.30) and (4.31), we will eventually obtain either of the following formula with respect to the result of the resolution of $@_1$.

$$(4.32)\quad \begin{array}{l} \lambda z'.p(\pi_1(s_f(z'))) : \\ \left(z' : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x, y) \end{array} \right] \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \left(v : \left[\begin{array}{l} \mathbf{farmer}(\pi_1 z') \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(\pi_1 z', y) \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \mathbf{beat}(\pi_1 z', \pi_1 \pi_2 v) \end{array}$$

$$(4.33)\quad \begin{array}{l} \lambda z'.p(\pi_1(s_f(z'))) : \\ \left(z' : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x, y) \end{array} \right] \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \left(v : \left[\begin{array}{l} \mathbf{farmer}(\pi_1 z') \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(\pi_1 z', y) \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \mathbf{beat}(\pi_1 z', \pi_1 \pi_2 \pi_2 (f(\pi_1 s_f(z')))) \end{array}$$

Although it is not immediately obvious what is going on here, in (4.32), the object position of **beat** refers to v, which is a universally quantified object. On the contrary, in (4.33), it refers to a particular entity which is obtained via f with respect to z'.

It is easy to show that (4.32) and the following classical semantic representation of universal quantifier are mutually deducible.

$$(4.34)\quad \left(z' : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x, y) \end{array} \right] \end{array} \right] \end{array} \right] \end{array} \right] \right) \to \mathbf{beat}(\pi_1 z', \pi_1 \pi_2 \pi_2 z')$$

On the contrary, from (4.33), we can prove the following proposition, which better reflects the existential reading.

$$(4.35)\quad \left(z' : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \rightarrow \left[\begin{array}{l} u : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(\pi_1 z', y) \end{array}\right] \end{array}\right] \\ \mathbf{beat}(\pi_1 z', \pi_1 u) \end{array}\right]$$

The semantic representation of *no* that account for both universal and existential reading can be given analogously based on (4.24).

$$(4.36)\quad \left[\begin{array}{l} f : \mathbf{M}_0 \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ (v : N(x)) \rightarrow V(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right]$$

Again, this is mutually deducible with the following formula.

$$(4.37)\quad \neg \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : N(x) \\ (v : N(x)) \rightarrow V(x) \end{array}\right] \end{array}\right]$$

Here again, if the @-term in V is resolved by using v, the semantic representation corresponds to the universal reading; if it is resolved by using u, it corresponds to the existential reading.[11]

4.5 Plural anaphora

In the previous section, we provided semantic representations of GQs such as *most* and the cardinal quantifier *three*. In this section, we will see their

[11] It is also possible to get the semantic representation of *most* for both universal and existential reading in a same manner. We will not further pursue the direction in this thesis.

behavior in the discourse, especially how they interact with plural pronouns.

4.5.1 Plural anaphora and dependent interpretation

Consider the following examples involving plural anaphora ((4.40) is due to Krifka (1996)).

(4.38) Every[1] boy received a[1] present.
They$_1$ opened it$_2$.

(4.39) Most[1] farmers own a[2] donkey.
They$_1$ love it$_2$.

(4.40) Three[1] students wrote a[2] paper.
They$_1$ sent it$_2$ to L&P.

Let us focus on a reading of (4.38) where *every boy* receives wide scope over *a present* (henceforth, the ∀–∃ reading). In that reading, the singular pronoun *it* in the second sentence refers to its antecedent in the nuclear scope of the quantifier in the first sentence: the second sentence of (4.38) can mean that every boy received a present and opened *the present he received*. By ∀–∃ reading, a similar interpretation applies to (4.39) and (4.40); the second sentence of (4.39) is understood to mean that most farmers own a donkey and love *the donkey he own*; the second sentence of (4.40) means that three students each wrote a paper and each student sent *the paper he wrote* to L&P.

Another thing to pay attention to is that, in all of the above examples, the singular pronoun *it* in the second sentence receives the interpretation that depends on the subject: *it* in (4.38) is interpreted as *the present he received*, where *he* refers to each of the subject *boy*; the same applies to

(4.39) and (4.40).

The source of such interpretations must be the interpretations of the first sentences, the quantificational sentences. For instance, the interpretation of *it* in (4.40) is possible because we know from the first sentence that there exists a paper for each of the three students. This suggests that the ∀–∃ reading of the first sentence induces a dependency relation between students and papers, which plays a crucial role in the interpretation of the singular pronoun in the second sentence.

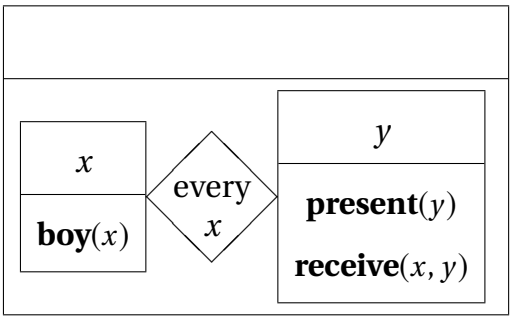
Therefore, analysis of anaphora such as (4.38), (4.39), and (4.40) should also account for this dependent interpretation.

4.5.2 Dependency relation

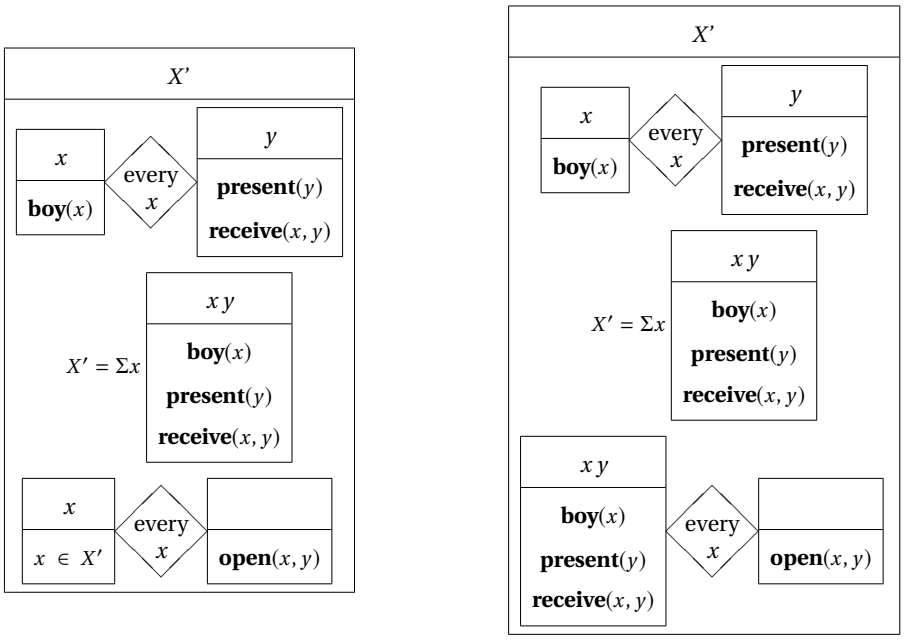
As the dependency interpretation is crucially involved in plural anaphora in general, it has been widely discussed in the literature (Kamp and Reyle, 1993; van den Berg, 1996a,b; Krifka, 1996; Nouwen, 2003; Brasoveanu, 2008).

In classical Discourse Representation Theory (DRT) (Kamp and Reyle, 1993), reference to a dependency relation is handled by using a copy mechanism. First, the first sentence in (4.38), *every boy received a present*, yields the following discourse representation structure (DRS).

(4.41)



This form of DRS is called *duplex condition* and used to represent a sentence involving a quantifier. The construction of this DRS triggers the op-



a. DRS for (4.38) before applying copy operation. b. DRS for (4.38) after applying copy operation.

Figure 4.3: DRS associated with (4.38).

eration called *abstraction*, which constructs a new plural discourse referent X' consisting of an object that satisfies the condition of x. The pronoun *they* refers to this X' and yields the DRS in Figure 4.3a, where universal quantification over X' takes place. In this DRS, however, there is no discourse referent which can be associated with singular y in **open**(x, y). In such a case, there is an option to apply a copy operation, which copies the conditions of x constituting X' to the restrictor part of the duplex condition. The corresponding DRS is given in Figure 4.3b. In this way, the singular variable y in **open**(x, y) can refer to each present associated with each boy. What plays an important role here is operations performed based on DRS construction.

Krifka (1996) tries to account for the phenomena without introducing an additional level of semantic representation such as DRSs. He proposes

an analysis based on an enriched assignment function called *parametrized sum individuals* based on the approach interpreting a sentence as a relation between assignment functions. Parametrized sum individuals are sets of pairs of an individual and a variable assignment associated with that individual. A possible instance of parametrized individuals for *every boy received a present* may have the following representation, where x and y are variables for boys b_i and presents p_i, respectively .

$$\langle x, \{\langle b_1, \{\langle y, p_1\rangle\}\rangle, \langle b_2, \{\langle y, p_2\rangle\}\rangle, \langle b_3, \{\langle y, p_3\rangle\}\rangle, ...\}\rangle$$

The individuals can be either singular or plural. Since individuals are followed by assignments associated with them, this structure captures dependency relations between objects. In the case of the distributive interpretation, each parametrized individual is independently evaluated against predicates. Thus, singular pronouns can be interpreted along each parametrized individual, which produces an effect of interpretation sensitive to the dependency relation. By introducing such a recursive notion of variable assignment, the theory keeps to interpret a sentence as a relation between assignment functions.

van den Berg (1996a,b) proposed to encode dependency relations by adopting *information states for plurals* in Dynamic Plural Logic. In this approach, formulas are interpreted relative to information states, which are sets of assignment functions, instead of to assignment functions as in standard Dynamic Predicate Logic (Groenendijk and Stokhof, 1991) and Krifka (1996)'s approach. A possible information state for *every boy received a present* may have the following structure, where x and y are again variables for boys b_i and presents p_i, respectively.

$$\{\{\langle x,\ b_1\rangle, \langle y,\ p_1\rangle\}, \{\langle x,\ b_2\rangle, \langle y,\ p_2\rangle\}, \{\langle x,\ b_3\rangle, \langle y,\ p_3\rangle\}, ...\}$$

When distribution over x is involved, predicates are evaluated against each assignment of information states. The assignment of new values takes place independently of each assignment function; thus, the variables introduced may be dependent on x. This is the source of dependency.

The main intuition behind these analyses, and also behind the analysis we will present here, is that the dependency relation is introduced by distributive reading. In general, interpretation of pronouns can be sensitive to linguistically introduced dependency relations between objects: in Chapter 2, we have seen that this dependency relation is involved in plural anaphora as well as quantificational subordination cases. The approaches mentioned above, however, account for only part of these phenomena. In DRT, the copy mechanism is triggered by the resolution of the plural pronoun *they*. Thus, it needs additional stipulation or operation to handle more general cases including quantificational subordination. The same can be said of approaches by Krifka (1996) and van den Berg (1996b). van den Berg's analysis can account for (4.42) where a subset relation allows reference to a dependency relation.

(4.42) Every boy will receive a^1 present. Every young boy will open it$_1$.

(4.43) Every boy will receive a^1 present. John will open it$_1$.

In general, however, a semantic link between the restrictor of the universal quantifier and the subject of the subsequent discourse is not limited to the subset relation, as in (4.43). Rather, the dependent interpretation involves a more general kind of inference, of which a semantic link in terms of subset relations is a special instance.

We will show that, in DTS, dependent types are readily provided as information source of such dependency between objects and can contribute to the context for pronoun resolution. In Chapter 2, we have ex-

plained that cases such as (4.42) and (4.43) can be explained in terms of Π-types in DTS setting. In Section 4.4, we have shown that the semantic representation of *every* on is analogous to Π-types, which captures the dependency relation.

Below, we provide a lexical entry of plural pronoun *they* and show that we can account for dependency interpretation in plural anaphora.

4.5.3 Plural pronoun

Let us take (4.40) as our working example, which is repeated here as (4.44).

(4.44) Three[1] students wrote a[2] paper.
They$_1$ sent it$_2$ to L&P.

The first sentence that involves *three* is represented as follows.

(4.45)
$$\left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n = 3 \\ \left[\begin{array}{ll} f : \mathbf{M}_k \rightarrow & \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{student}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} v : \mathbf{paper}(y) \\ \mathbf{write}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{injection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

Here, f is a function that returns tuples. Each of the tuples involves two entities, one is a student and the other is a paper, that come together with the proof that the student writes the paper. In other words, (4.45) represents the dependency between students and papers.

The pronoun *they* in the second sentence is represented by an underspecified term, as other anaphoric expressions are. Singular pronouns

such as *it* or *he* are represented in terms of @-terms annotated with type **entity**, which ensures that the term will be eventually replaced with some entity in the context. In contrast, the semantic representation of *they* is defined as referring to multiple entities in the context.

Therefore, we want to have a type as follows to annotate the @-term:

$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq 2 \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \mathbf{entity} \\ \mathbf{injection}\,(f) \end{array}\right] \end{array}\right] \end{array}\right] \tag{4.46}$$

Let us write this type $\mathcal{PL}$ for short. With $\mathcal{PL}$, we can give a semantic representation of *they* as follows.

$$\lambda v.\,\left(z : \mathbf{M}_{\pi_1(@_i :: \mathcal{PL})}\right) \rightarrow v((\pi_1\pi_2\pi_2(@_i :: \mathcal{PL}))z)\,, \tag{4.47}$$

Here, $\pi_1(@_i :: \mathcal{PL})$ is a cardinal number and $\pi_1\pi_2\pi_2(@_i :: \mathcal{PL})$ is a function $\mathbf{M}_k \rightarrow \mathbf{entity}$. The semantic representation of *they* analyzed as introducing distributive interpretation.[12]

By assigning CCG category $S/(S\backslash NP)$ to *they*, it is straightforward to obtain the following semantic representation for the second sentence of (4.44). The Underspecified term $@_1$ is introduced by *they* and $@_2$ by *it*.

$$\left(z : \mathbf{M}_{\pi_1(@_1 :: \mathcal{PL})}\right) \rightarrow \mathbf{sendItToL\&P}((\pi_1\pi_2\pi_2(@_1 :: \mathcal{PL}))z, @_2 :: \mathbf{entity}) \tag{4.48}$$

After combining two sentences by the progressive conjunction, anaphora resolution takes place. From the proof term S_1 of the first sentence (4.45) , one can construct the following proof term for $@_1$:

$$\langle 3,\ \mathbf{p}_{3\geq 2},\ \lambda x.\pi_1((\pi_1\pi_2\pi_2 S_1)x),\ \mathbf{p}_{\mathbf{injection}(\lambda x.\pi_1((\pi_1\pi_2\pi_2(S_1))x))}\rangle \tag{4.49}$$

[12] Some may prefer to introduce distributivity separately from *they*. As the sentence needs to receive distributive reading in any way, it is not harmless to assume this lexical entry with respect to our current purpose.

Here, $\mathbf{p}_{3\geq 2}$ is a proof that *3 is greater or equal to 2*. The term $\pi_1\pi_2\pi_2 S_1$ is a function introduced by the first sentence, who has type $\mathbf{M}_3 \to (x : \mathbf{entity}) \times A$ where A is the complex Σ-type. Recall that, given $f : \mathbf{M}_k \to (x : \mathbf{entity}) \times A$ and $i : \mathbf{injection}_{\pi_1}(f)$, we can obtain $\lambda x.\pi_1 f(x) : \mathbf{M}_k \to \mathbf{entity}$ and $i' : \mathbf{injection}\,(\lambda x.\pi_1 f(x))$ (see Section 4.1.5). The third and fourth component of (4.49) are constructed from $\pi_1\pi_2\pi_2(S_1)$ in this way.

When $@_1$ is resolved with the proof term shown in (4.49), one obtains the following intermediate semantic representation.

$$
(4.50)\quad
\left[\begin{array}{l}
S_1 : \left[\begin{array}{l}
n : \mathbf{Nat} \\
\left[\begin{array}{l}
n = 3 \\
\left[\begin{array}{ll}
f : \mathbf{M}_k \to & \left[\begin{array}{l}
x : \mathbf{entity} \\
\left[\begin{array}{l}
u : \mathbf{student}(x) \\
\left[\begin{array}{l}
y : \mathbf{entity} \\
\left[\begin{array}{l}
v : \mathbf{paper}(y) \\
\mathbf{write}(x,y)
\end{array}\right]
\end{array}\right]
\end{array}\right]
\end{array}\right] \\
\mathbf{injection}_{\pi_1}(f)
\end{array}\right]
\end{array}\right]
\end{array}\right] \\
(z : \mathbf{M}_3) \to \mathbf{sendItToL\&P}(\pi_1((\pi_1\pi_2\pi_2 S_1)z), @_2 :: \mathbf{entity})
\end{array}\right]
$$

Therefore, by resolving $@_2$ with $\pi_1\pi_2\pi_2((\pi_1\pi_2\pi_2 S_1)z)$, one can obtain the fully-specified representation of (4.44). After the anaphora resolution, the first and second arguments of **sendItToL&P** both involve $(\pi_1\pi_2\pi_2 S_1)z$. This is a proof that *a student x write a paper y*. As the first and second arguments of **sendItToL&P** refer to the entities corresponding to x and y, respectively, the two arguments satisfy **write** which is witnessed by the proof $(\pi_1\pi_2\pi_2 S_1)z$. In this way, the semantic representation captures the dependency relation between students and papers.

Note also that, as we have discussed in Section 4.4, the universally-quantified entities in the second sentence are only those who were men-

tioned in the first sentence, whose cardinality is three. This is because $\mathcal{PL}$ is to look for not entities themselves but a function introduced by the preceding sentence. As the quantifier *three* introduces a function from $\mathbf{M}_3$ to the Σ-type, we are able to refer to only those three students as is intended.

The same analysis applies to the case where the quantifier of the first sentence introduces $\mathbf{bijection}_{\pi_1}$. Since $\mathbf{bijection}_{\pi_1}$ is defined as a conjunction of $\mathbf{injection}_{\pi_1}$ and $\mathbf{surjection}_{\pi_1}$, one can construct an object of $\mathcal{PL}$ also from such sentences.

An advantage of the proposed DTS analysis is that dependent types are readily provided as structures that can capture dependency between objects: from the proof p of $(x : A) \times B$, we can obtain $\pi_1 p : A$, which is accompanied by the witness $\pi_2 p : B(\pi_1 p)$; from the proof f of $(x : A) \to B$, we can obtain the witness $f(p) : B(p)$ for any $p : A$. Thus, by following the standard dynamic conjunction operation and anaphora resolution procedure as proof search, those dependency relation encoded as proofs can naturally contribute to anaphora resolution in general, including plural anaphora and quantificational subordination.

Chapter 5

Quantifier and Inference

In this chapter, we will show that the semantic representations of quantifiers defined in the previous chapter capture their inferential properties.

5.1 Entailment of quantified sentences

5.1.1 Inferential properties

As we have mentioned in the earlier chapter, quantifiers exhibit various inference patterns in relation to the sets they associate with. Conservativity is a general property that all natural language quantifiers have. It is model-theoretically formulated as follow (see, e.g., Peters and Westerståhl, 2006).

Quantifier Q is *conservative* if and only if, for all universe M and $A, B \subseteq M$, $Q_M(A, B) \Leftrightarrow Q_M(A, A \cap B)$

Monotonicity is another well-known phenomenon. There are four types of monotonicity: *right upward* (MON $\uparrow$), *right downward* (MON $\downarrow$), *left up-*

ward (↑ MON), and *left downward* (↓ MON). Here we repeat the definition of right upward monotonicity.

Quantifier Q is *Right upward monotone* (MON↑) if and only if, for all M, $A \subseteq M$, and $B \subseteq B' \subseteq M$, $Q_M(A,B)$ implies $Q_M(A,B')$.

For our purpose, these properties need to be defined from both proof-theoretic and dynamic perspective (Kanazawa, 1994). We first consider conservativity and right upward/downward monotonicity. We can formulate them as entailments in dependent type theory.

Definition 14 (Conservativity)**.** A quantifier **Q** is conservative if both judgements

$$\Gamma, p : \mathbf{Q}(A,B) \vdash \mathbf{Q}(A,(u:A)\times B)\ \mathit{true}$$
$$\Gamma, p : \mathbf{Q}(A,(u:A)\times B) \vdash \mathbf{Q}(A,B)\ \mathit{true}$$

hold, where

$$\Gamma \vdash A : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity},\ u : A(x) \vdash B : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma \vdash \mathbf{Q}(A,B) : \mathbf{type}$$

Definition 15 (Right upward monotonicity)**.** A quantifier **Q** is right upward monotone if the judgement

$$\Gamma,\ p : \mathbf{Q}(A,B),\ q : \big(x : \mathbf{entity}\big) \to (u : A(x)) \to B(x) \to B'(x) \vdash \mathbf{Q}(A,B')\ \mathit{true}$$

holds, where

$$\Gamma \vdash A : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity},\ u : A(x) \vdash B : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity},\ u : A(x) \vdash B' : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma \vdash \mathbf{Q}(A,B) : \mathbf{type}$$

In our proof-theoretic setting, the meaning of a sentence is justified by the entailment of its semantic representation. In this section, we will provide a proof that these entailment relations hold for the semantic representation of quantifiers given in the previous section.

When human judge entailment relations between sentences, we focus on the particular reading where the interpretation of anaphoric expressions is already fixed. This means that anaphora resolution has already taken place, in other words, the judgement does not contain any @-term. Therefore, for some strong determiner Q, we can assume that there exists some proof in the context which can resolve the existential presupposition of both the premise and the consequence. Also, as shown in the above definitions, the scope B can contain the proof term u of the restrictor A as a free variable, if B contained an @-term that has been resolved by the proof of A. This is the dynamic setting which has been also explored in the context of DPL (Kanazawa, 1994).

When the quantifier Q is either *most*, *at least n*, *at most n*, or *exactly n*, our semantic representations of quantified sentences, which correspond to $\exists$-reading, can be schematically written as follows.

$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} r : \mathcal{R}(k, \mathcal{N}) \\ \left[\begin{array}{ll} f : \mathbf{M}_k \rightarrow & \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1} f & \end{array}\right] \end{array}\right] \end{array}\right] \tag{5.1}$$

We call this type $Q^{\exists}$. Here, $\mathcal{N}$ is a natural number and $\mathcal{R}$ is a binary relation between natural numbers. They are determined with respect to

type	$\mathcal{N}$	$\mathcal{R}$	quantifiers	monotonicity
$Q^{\exists}_{most}$	$*$	$\geq$	most	MON↑
$Q^{\exists}_{atLeast}$	n	$\geq$	at least n	↑MON↑
$Q^{\exists}_{atMost}$	n	$\leq$	at most n	↓MON↓
$Q^{\exists}_{exactly}$	n	$=$	exactly n	neither

Table 5.1: Relation between the quantifier types and inference. If the number $\mathcal{N}$ changes depending on the restrictor's cardinality, it is indicated by $*$.

the choice of the quantifiers. Table 5.1 summarize the relations between the quantifiers and the combinations of $\mathcal{R}$ and $\mathcal{N}$ used in their semantic representations. All of them are *conservative*, but they exhibit a different pattern of right monotonicity: when $\mathcal{R}$ is the $\geq$-relation, the quantifiers are monotone increasing; if it is the $\leq$-relation, the quantifiers are monotone decreasing. The choice of the specific number $\mathcal{N}$ is, for instance, 3 for *at least three*. For the quantifier *most*, $\mathcal{N}$ is a natural number that corresponds to more than half of the restrictor's cardinality. As we have explained in the previous chapter, the number is obtained through the presupposition resolution.

5.2 Conservativity

The proof of conservativity is straightforward, as the following two are mutually deducible for any A, B : **type** regardless of the choice of the particular k.

$$\left[\begin{array}{l} f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \dashv\vdash \left[\begin{array}{l} g : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} v : A(x) \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(g) \end{array}\right]$$

Thus, we can always construct a proof of $\mathbf{Q}(A, (u : A) \times B)$ from a proof of $\mathbf{Q}(A, B)$, and vice versa.

We will prove only the case of type $Q^{\exists}$. The theorem on their conservativity is formulated as below.

Theorem 2: Conservativity (Type $Q^{\exists}$ quantifiers)

$$\left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} r : \mathcal{R}(k, \mathcal{N}) \\ \left[\begin{array}{l} f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \end{array}\right] \end{array}\right] \mathit{true} \dashv\vdash \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} r : \mathcal{R}(k, \mathcal{N}) \\ \left[\begin{array}{l} g : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} v : A(x) \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(g) \end{array}\right] \end{array}\right] \end{array}\right] \mathit{true}$$

holds under the context Γ, where

$$\Gamma \vdash A : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity}, u : A(x) \vdash B : \mathbf{entity} \to \mathbf{type}$$
$$v \notin fv(A) \cup fv(B)$$

Proof. The proof of the left-to-right direction is given as follows. When the type on the left side is *inhabited*, there exists a proof of this type. We write it M.

From M, we can extract its part by ΣE. We write the letters shown below

in the left hand side for the proof terms in the right hand side.

$$k \equiv \quad \pi_1 M : \mathbf{Nat} \tag{5.2}$$

$$r \equiv \quad \pi_1\pi_2 M : \mathcal{R}(k, \mathcal{N}) \tag{5.3}$$

$$f \equiv \quad \pi_1\pi_2\pi_2 M : \mathbf{M}_k \to \big(x : \mathbf{entity}\big) \times (u : A(x)) \times B(x) \tag{5.4}$$

$$b_f \equiv \quad \pi_2\pi_2\pi_2 M : \mathbf{bijection}_{\pi_1}(f) \tag{5.5}$$

Let us begin with constructing the function g. Assume $y : \mathbf{M}_k$. By applying ΠE to y and f, we get

$$f(y) : \big(x : \mathbf{entity}\big) \times (u : A(x)) \times B(x) \tag{5.6}$$

By applying ΣE several times, we get each of the following terms.

$$\pi_1 f(y) : \mathbf{entity} \tag{5.7}$$

$$\pi_2 f(y) : \{\, (u : A(x)) \times B(x) \,\}_{[\pi_1 f(y)/x]} \tag{5.8}$$

$$\pi_1\pi_2 f(y) : \{\, A(x) \,\}_{[\pi_1 f(y)/x]} \tag{5.9}$$

Here, the notation $\{\, A(x) \,\}_{[a/x]}$ means that every occurrence of the free variable x in A is substituted with a.

By ΣI with (5.9) and (5.8), we get

$$\langle \pi_1\pi_2 f(y), \pi_2 f(y) \rangle : \left\{ \begin{bmatrix} v : A(x) \\ (u : A(x)) \times B(x) \end{bmatrix} \right\}_{[\pi_1 f(y)/x]} \tag{5.10}$$

By ΣI together with (5.7), we obtain

$$\langle \pi_1 f(y), \langle \pi_1\pi_2 f(y), \pi_2 f(y) \rangle \rangle : \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} v : A(x) \\ (u : A(x)) \times B(x) \end{bmatrix} \end{bmatrix}, \tag{5.11}$$

from which, we get the following by discharging the assumption.

$$\lambda y.\langle \pi_1 f(y), \langle \pi_1\pi_2 f(y), \pi_2 f(y) \rangle \rangle : \mathbf{M}_k \to \begin{bmatrix} x : \mathbf{entity} \\ \begin{bmatrix} v : A(x) \\ (u : A(x)) \times B(x) \end{bmatrix} \end{bmatrix} \tag{5.12}$$

We write this function g.

Next, we prove that $\mathbf{injection}_{\pi_1}(g)$ and $\mathbf{surjection}_{\pi_1}(g)$ hold.

Given $y : \mathbf{M}_k$, $y' : \mathbf{M}_k$, if $\pi_1 g(y) = \pi_1 g(y')$ holds, $\pi_1 f(y) = \pi_1 f(y')$ holds because $\lambda y.\pi_1 g(y) \equiv \lambda y.\pi_1 f(y)$ holds according to (5.12). As $\mathbf{injection}_{\pi_1}(f)$ is true by (5.5), $y = y'$ holds. Thus, we can conclude that $\mathbf{injection}_{\pi_1}(g)$ holds.

To prove $\mathbf{surjection}_{\pi_1}(g)$, assume $z : (x : \mathbf{entity}) \times (v : A(x)) \times (u : A(x)) \times B(x)$. we will prove that $\begin{bmatrix} y : \mathbf{M}_k \\ \pi_1 z = \pi_1 g(y) \end{bmatrix}$ *true*, namely, $\begin{bmatrix} y : \mathbf{M}_k \\ \pi_1 z = \pi_1 f(y) \end{bmatrix}$ *true* holds.

From z, we obtain:

$$\pi_1 z : \mathbf{entity} \tag{5.13}$$

$$\pi_2 z : \{ (v : A(x)) \times (u : A(x)) \times B(x) \}_{[\pi_1 z/x]} \tag{5.14}$$

$$\pi_2 \pi_2 z : \{ (u : A(x)) \times B(x) \}_{[\pi_1 z/x][\pi_1 \pi_2 z/v]} \tag{5.15}$$

As $v \notin fv(A) \cup fv(B)$, $\{ (u : A(x)) \times B(x) \}_{[\pi_1 z/x][\pi_1 \pi_2 z/v]} \equiv \{ (u : A(x)) \times B(x) \}_{[\pi_1 z/x]}$. Thus, by applying ΣI with (5.13), we obtain

$$\langle \pi_1 z, \pi_2 \pi_2 z \rangle : \begin{bmatrix} x : \mathbf{entity} \\ (u : A(x)) \times B(x) \end{bmatrix} \tag{5.16}$$

From b_f of (5.5), we get $\pi_2 b_f : \left(z' : \begin{bmatrix} x : \mathbf{entity} \\ (u : A(x)) \times B(x) \end{bmatrix} \right) \to \begin{bmatrix} y : \mathbf{M}_k \\ \pi_1 z' = \pi_1 f(y) \end{bmatrix}$.

Thus, by applying ΠE with $\pi_2 b_f$ and (5.16), we obtain our goal $\begin{bmatrix} y : \mathbf{M}_k \\ \pi_1 z = \pi_1 f(y) \end{bmatrix}$.

We can prove the right-to-left direction in an analogous way.

End of proof.

Note that the proof above is independent of the types such as **entity** and $\mathbf{M}_k$.

In the proof above, we are careful to handle the variables when proving $(v : A(x)) \times (u : A(x)) \times B(x)$ from $(u : A(x)) \times B(x)$ and vice versa. This is because there may be a dependency between A and B, as anaphora resolution has already taken place. Specifically, the proof above can account for the conservativity of a sentence involving anaphora.

(5.1) Most farmers who own a^1 donkey beat it$_1$.
⇔ Most farmers who own a donkey are farmers who own a^1 donkey and beat it$_1$.

It is also straightforward to prove a theorem where B depends on v not on u, in other words, only v occurs free in B. Such a case corresponds to the following entailment.

(5.2) Most farmers who own a^1 donkey beat it$_1$.
⇔ Most farmers who own a^1 donkey are farmers who own a donkey and beat it$_1$.

5.3 Proofs of right monotonicity

5.3.1 Overview of the proofs

The proof of right monotonicity is more tricky. Let us first consider right upward monotonicity.

Our intuition behind the right upward monotonicity of *most* is that the originally established relationship between A and B is preserved even if B is enlarged to B'. In other words, we are interested in only the preservation of the relation between those numbers. As we adopt bijection in the semantic representation, however, we need to construct a new bijection that extends B to B', which gives the corresponding specific numbers.

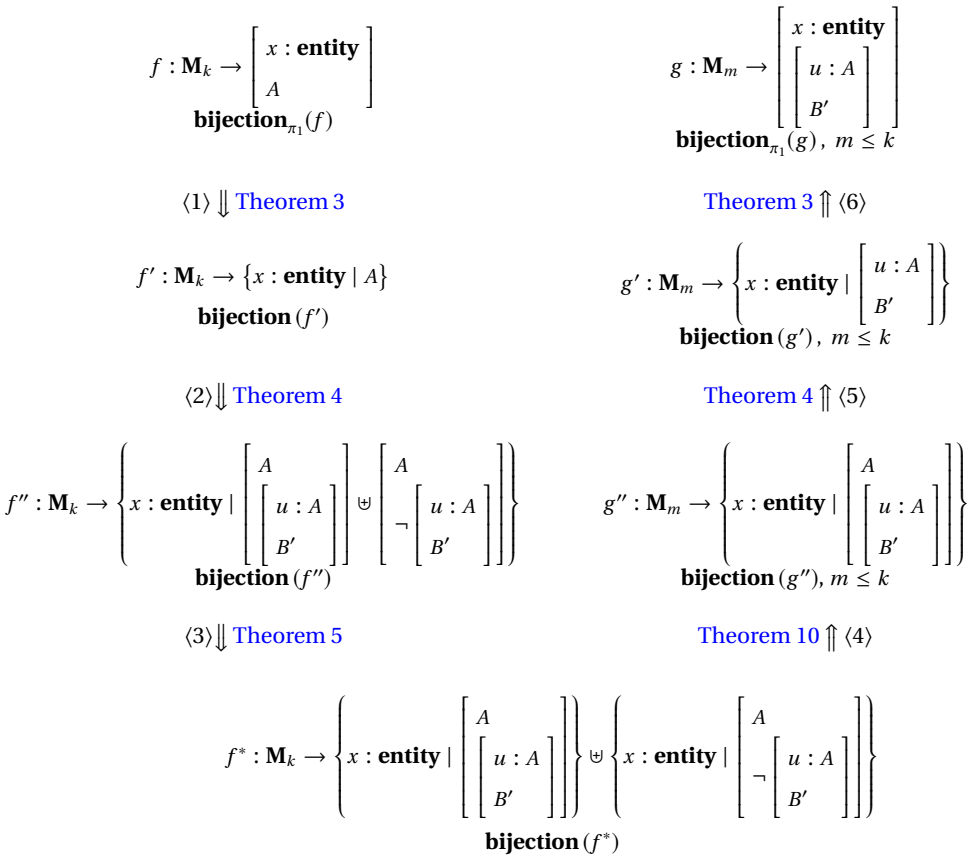


Figure 5.1: Diagram showing the proof steps of Theorem 11.

Therefore, we will carry out the proof in two steps. In the first step, we assume the finiteness of the restrictor and show that its part, $(x : \mathbf{entity}) \times (u : A(x)) \times B'(x)$, is also finite with respect to **entity**. This proof will give the cardinal number of $(x : \mathbf{entity}) \times (u : A) \times B'$. In the next step, we will prove that the number satisfies the expected large/small relation with respect to the restrictor.

The proof process for the first step is illustrated in Figure 5.1.

The step ⟨1⟩, Theorem 3, concerns the mutual conversion between $\mathbf{bijection}_{\pi_1}$ and **bijection**. For this theorem, we introduce subset types (Nordström et al., 1990; Jacobs, 1998). We use the subset type for the transition

Formation rule

$$\frac{\begin{array}{c}\overline{x : A}^{\,\mathrm{i}} \\ \vdots \\ A : \mathbf{type} \quad B(x) : \mathbf{type}\end{array}}{\{x : A \mid B(x)\} : \mathbf{type}}\ \{|\}F, \mathrm{i}$$

Introduction rule

$$\frac{M : A \quad N : B[M/x]}{\mathsf{i}_B(M) : \{x : A \mid B(x)\}}\ \{|\}I$$

Elimination rule (standard)

$$\frac{\begin{array}{cc}\overline{x : A}^{\,\mathrm{i}} & \overline{y : B(x)}^{\,\mathrm{i}} \\ & \vdots \\ M : \{x : A \mid B(x)\} & N : C(x)\end{array}}{N[\mathsf{o}_B(M)/x]}\ \{|\}E, \mathrm{i} \quad \text{where } y \notin fv(N) \cup fv(C)$$

Elimination rules we adopt

$$\frac{M : \{x : A \mid B(x)\}}{\mathsf{o}_B(M) : A}\ \{|\}E \qquad \frac{M : \{x : A \mid B(x)\}}{?(M) : B[\mathsf{o}_B(M)/x]}\ \{|\}E_?$$

Computation rules

$$\mathsf{i}_B(\mathsf{o}_B(M)) \to_\beta M \qquad \mathsf{o}_B(\mathsf{i}_B(M)) \to_\beta M$$

Figure 5.2: Inference rules of subset type.

from the Σ-type to the type consisting only of entities. Figure 5.2 shows the inference rules. The introduction rule looks similar to that of the Σ-type. Unlike the Σ-type, the subset type $\{x : A \mid B(x)\}$ consists only of A's elements satisfying B, where B is often called *specification*. A proof of the specification is used only to confirm the membership of the subset and is forgotten afterward. This property is reflected in the standard elimination rule. According to the standard elimination rule, when z is an element of $\{x : A \mid B(x)\}$, we cannot use proof of $B(z)$, even though it is obvious to

hold.

Therefore, we use the abnormal elimination rule here. The elimination rule on the left is derived from the standard one. The elimination rule on the right is adopted for using the proof of $B(z)$. As we can not know the precise proof term, we express it in terms of a constructor. We will crucially use this rule in Theorem 3 and Theorem 5.[1]

Theorem 4 concerns the inference on the specification of subset type. Theorem 10 states that if the disjoint union of types is finite, then a type is also finite. In all the proof steps shown in Figure 5.1, we assume that for arbitrary P : **entity** $\to$ **type**, P is a *decidable predicate*; that is, for every x : **entity**, it is either $P(x)$ or $\neg P(x)$.

We will give the proofs of these theorems in the following sections and then prove that the quantifiers we have defined satisfy the right monotonicity property as intended in Section 5.4.

5.3.2 Theorem: convertibility of bijection and bijection$_{\pi_1}$

Theorem 3

(1) For any P : **entity** $\to$ **type** and k : **Nat**,

$$\Gamma,\ f : \mathbf{M}_k \to \left[\begin{array}{l} x : \textbf{entity} \\ P(x) \end{array}\right],\ b_f : \textbf{bijection}_{\pi_1}(f) \vdash \left[\begin{array}{l} f' : \mathbf{M}_k \to \{x : \textbf{entity} \mid P(x)\} \\ \textbf{bijection}\,(f') \end{array}\right] \mathit{true}$$

[1] This rule may disrupt the harmony of the system, as an introduction immediately followed by an elimination cannot give exactly the same proof of $B(z)$, which means the system does not satisfy the normalization property. A detailed investigation of the impact on the system is left for future work.

(2) For any P : **entity** $\to$ **type** and k : **Nat**,

$$\Gamma,\ f' : \mathbf{M}_k \to \left\{x : \mathbf{entity} \mid P(x)\right\},\ b'_f : \mathbf{bijection}\,(f') \vdash \left[\begin{array}{l} f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(f) \end{array}\right] \mathit{true}$$

Proof. (1) The function $f' : \mathbf{M}_k \to \left\{x : \mathbf{entity} \mid P(x)\right\}$ is constructed as follows.

$$\dfrac{\dfrac{\dfrac{f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right] \quad \dfrac{}{y : \mathbf{M}_k}\,1}{f(y) : \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right]}\,\Pi E \quad \dfrac{f : \mathbf{M}_k \to \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right] \quad \dfrac{}{y : \mathbf{M}_k}\,1}{f(y) : \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right]}\,\Pi E}{\pi_1 f(y) : \mathbf{entity}}\,\Sigma E \quad \dfrac{}{\pi_2 f(y) : P(\pi_1 f(y))}\,\Sigma E}{\dfrac{\mathsf{i}_P(\pi_1 f(y)) : \left\{x : \mathbf{entity} \mid P(x)\right\}}{\lambda y.\mathsf{i}_P(\pi_1 f(y)) : \mathbf{M}_k \to \left\{x : \mathbf{entity} \mid P(x)\right\}}\,\Pi I, 1}\,\{|\}I$$

Next, we prove that it is an injection. Assume $y : \mathbf{M}_k$, $y' : \mathbf{M}_k$, and $u : \mathsf{i}_P(\pi_1 f(y)) =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\pi_1 f(y'))$.

We have

$$\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y)))) : \mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y))) =_{\mathbf{entity}} \mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y))). \tag{5.17}$$

By substitution with u, we get[2]

$$\mathbf{p}_{15}(\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y))))) : \mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y))) =_{\mathbf{entity}} \mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y'))), \tag{5.18}$$

which is computed to

$$\mathbf{p}_{15}(\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 f(y))))) : \pi_1 f(y) =_{\mathbf{entity}} \pi_1 f(y') \tag{5.19}$$

[2] By the bold face $\mathbf{p}_i$ with subscripted index, we indicate that we use the theorems or derived rules presented in the appendix.

As f satisfies $\mathbf{injection}_{\pi_1}(f)$, we conclude that $y = y'$ holds.

Next, we will prove that the function is a surjection. We will prove $\left[\begin{array}{l} y : \mathbf{M}_k \\ z =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\pi_1 f(y)) \end{array}\right]$ by assuming $z : \{x : \mathbf{entity} \mid P(x)\}$. From z, we obtain $\mathsf{o}_P(z) : \mathbf{entity}$ and $?(z) : P(\mathsf{o}_P(z))$ by subset elimination. Hence, by ΣI, we obtain

$$\langle \mathsf{o}_P(z), ?(z)\rangle : \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right] \tag{5.20}$$

As f satisfies $\mathbf{surjection}_{\pi_1}(f)$, we have

$$\pi_2 b_f : \left(z : \left[\begin{array}{l} x : \mathbf{entity} \\ P(x) \end{array}\right]\right) \to \left[\begin{array}{l} y : \mathbf{M}_k \\ \pi_1 z =_{\mathbf{entity}} \pi_1 f(y) \end{array}\right]. \tag{5.21}$$

By applying ΠE to (5.20) and (5.21), we get

$$(\pi_2 b_f)(\langle \mathsf{o}_P(z), ?(z)\rangle) : \left[\begin{array}{l} y : \mathbf{M}_k \\ \mathsf{o}_P(z) =_{\mathbf{entity}} \pi_1 f(y) \end{array}\right]. \tag{5.22}$$

We abbreviate the term obtained in (5.22) to w. By applying ΣE to (5.22), we obtain

$$\pi_1(w) : \mathbf{M}_k \tag{5.23}$$

$$\pi_2(w) : \mathsf{o}_P(z) =_{\mathbf{entity}} \pi_1 f(\pi_1(w)). \tag{5.24}$$

Now, we have

$$\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(z))) : \mathsf{i}_P(\mathsf{o}_P(z)) =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\mathsf{o}_P(z)) \tag{5.25}$$

By the substitution with (5.24), we get

$$\begin{array}{l} \mathbf{p}_{15}(\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(z)))) \\ \qquad : \mathsf{i}_P(\mathsf{o}_P(z)) =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\pi_1 f(\pi_1(w))) \end{array}, \tag{5.26}$$

which is computed to

$$\begin{array}{l}\mathbf{p}_{15}(\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(z))))\\ \qquad : z =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\pi_1 f(\pi_1(w)))\end{array}, \tag{5.27}$$

Thus, by applying ΣI to (5.23) and (5.27), we finally get the following proof.

$$\langle \pi_1(w), \mathbf{p}_{15}(\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(z))))\rangle : \begin{bmatrix} y : \mathbf{M}_k \\ z =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\pi_1 f(y)) \end{bmatrix}$$

(2) The proof for the other direction is analogous to (1). The function $f : \mathbf{M}_k \to \begin{bmatrix} x : \mathbf{entity} \\ P(x) \end{bmatrix}$ is constructed as follows.

$$\dfrac{\dfrac{\dfrac{f' : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \quad \overline{y : \mathbf{M}_k}^{\,1}}{f'(y) : \{x : \mathbf{entity} \mid P(x)\}}\,\Pi E}{\mathsf{o}_P(f'(y)) : \mathbf{entity}}\,\{|\}E \quad \dfrac{\dfrac{f' : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \quad \overline{y : \mathbf{M}_k}^{\,1}}{f'(y) : \{x : \mathbf{entity} \mid P(x)\}}\,\Pi E}{?(f'(y)) : P(\mathsf{o}_P(f'(y)))}\,\{|\}E_?}{\langle \mathsf{o}_P(f'(y)), ?(f'(y))\rangle : \begin{bmatrix} x : \mathbf{entity} \\ P(x) \end{bmatrix}}\,\Sigma I$$

$$\lambda y.\langle \mathsf{o}_P(f'(y)), ?(f'(y))\rangle : \mathbf{M}_k \to \begin{bmatrix} x : \mathbf{entity} \\ P(x) \end{bmatrix}\,\Pi I, 1$$

We show that $\mathbf{injection}_{\pi_1}(f)$ holds. Assume $y : \mathbf{M}_k$, $y' : \mathbf{M}_k$, and $u : \mathsf{o}_P(f'(y)) =_{\mathbf{entity}} \mathsf{o}_P(f'(y'))$.

We have

$$\begin{array}{l}\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(f'(y))))\\ \qquad : \mathsf{i}_P(\mathsf{o}_P(f'(y))) =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\mathsf{o}_P(f'(y)))\end{array} \tag{5.28}$$

By substitution with u, we get

$$\begin{array}{l}\mathbf{p}_{15}(\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(f'(y)))))\\ \qquad : \mathsf{i}_P(\mathsf{o}_P(f'(y))) =_{\{x:\mathbf{entity}|P(x)\}} \mathsf{i}_P(\mathsf{o}_P(f'(y'))),\end{array} \tag{5.29}$$

which is computed to

$$\mathbf{p}_{15}(\mathsf{refl}_{\{x:\mathbf{entity}|P(x)\}}(\mathsf{i}_P(\mathsf{o}_P(f'(y))))) : f'(y) =_{\{x:\mathbf{entity}|P(x)\}} f'(y'). \tag{5.30}$$

As f' is an injection, we conclude that $y = y'$ holds.

Next, we show that $\mathbf{surjection}_{\pi_1}(f)$ holds. We prove $\begin{bmatrix} y : \mathbf{M}_k \\ \pi_1 z = \mathsf{o}_P(f'(y)) \end{bmatrix}$ by assuming $z : \begin{bmatrix} x : \mathbf{entity} \\ P(x) \end{bmatrix}$.

From the assumption, we get $\pi_1 z : \mathbf{entity}$ and $\pi_2 z : P(\pi_1 z)$ by ΣE. Thus, by subset introduction, we obtain

$$\mathsf{i}_P(\pi_1 z) : \left\{x : \mathbf{entity} \mid P(x)\right\}. \tag{5.31}$$

As $\mathbf{surjection}\,(f')$ holds, we have

$$\pi_2 b_f : \left(z : \left\{x : \mathbf{entity} \mid P(x)\right\}\right) \to \begin{bmatrix} y : \mathbf{M}_k \\ z = f'(y) \end{bmatrix}. \tag{5.32}$$

By applying ΠE to (5.31) and (5.32), we get

$$(\pi_2 b_f)(\mathsf{i}_P(\pi_1 z)) : \begin{bmatrix} y : \mathbf{M}_k \\ \mathsf{i}_P(\pi_1 z) = f'(y) \end{bmatrix} \tag{5.33}$$

We abbreviate the obtained proof term to w'. By applying ΣE, we obtain

$$\pi_1 w' : \mathbf{M}_k \tag{5.34}$$

$$\pi_2 w' : \mathsf{i}_P(\pi_1 z) = f'(\pi_1 w') \tag{5.35}$$

Now, we have

$$\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 z))) : \mathsf{o}_P(\mathsf{i}_P(\pi_1 z)) =_{\mathbf{entity}} \mathsf{o}_P(\mathsf{i}_P(\pi_1 z)) \tag{5.36}$$

Thus, by substitution with (5.35), we get

$$\mathbf{p}_{15}(\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 z)))) : \mathsf{o}_P(\mathsf{i}_P(\pi_1 z)) =_{\mathbf{entity}} \mathsf{o}_P(f'(\pi_1 w')), \tag{5.37}$$

which is computed to

$$\mathbf{p}_{15}(\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 z)))) : \pi_1 z =_{\mathbf{entity}} \mathsf{o}_P(f'(\pi_1 w')). \tag{5.38}$$

Therefore, by applying ΣI to (5.34) and (5.38), we finally get the following proof.

$$\langle \pi_1 w', \mathbf{p}_{15}(\mathsf{refl}_{\mathbf{entity}}(\mathsf{o}_P(\mathsf{i}_P(\pi_1 z))))\rangle : \left[\begin{array}{l} y : \mathbf{M}_k \\ \pi_1 z = \mathsf{o}_P(f'(y)) \end{array}\right]$$

End of proof.

5.3.3 Theorem: subset types with equivalent specification

Theorem 4

For any $P,\ Q : \mathbf{entity} \to \mathbf{type}$ and $k : \mathbf{Nat}$,

$$\Gamma,\ f : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\},\ b_f : \mathbf{bijection}(f),\ t : (x : \mathbf{entity}) \to (P(x) \leftrightarrow Q(x)) \vdash \left[\begin{array}{l} g : \mathbf{M}_k \to \{x : \mathbf{entity} \mid Q(x)\} \\ \mathbf{bijection}(g) \end{array}\right] \ true$$

Proof. We begin with constructing a function of $\mathbf{M}_k \to \{x : \mathbf{entity} \mid Q(x)\}$. Let t_1 be of type $(x : \mathbf{entity}) \to P(x) \to Q(x)$ and t_2 be of type $(x : \mathbf{entity}) \to Q(x) \to P(x)$. We can construct the function as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{f : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \quad \overline{y : \mathbf{M}_k}^{\,1}}{f(y) : \{x : \mathbf{entity} \mid P(x)\}}\,\Pi E}{\mathsf{o}_P(f(y)) : \mathbf{entity}}\,\{|\}E \quad \dfrac{t_1 : (x : \mathbf{entity}) \to P(x) \to Q(x)}{t_1(\mathsf{o}_P(f(y))) : P(\mathsf{o}_P(f(y))) \to Q(\mathsf{o}_P(f(y)))}\,\Pi E \quad \dfrac{f(y) : \{x : \mathbf{entity} \mid P(x)\}}{?(f(y)) : P(\mathsf{o}_P(f(y)))}\,\{|\}E_?}{(t_1(\mathsf{o}_P(f(y))))(?(f(y))) : Q(\mathsf{o}_P(f(y)))}\,\Pi E}{\mathsf{i}_Q(\mathsf{o}_P(f(y))) : \{x : \mathbf{entity} \mid Q(x)\}}\,\{|\}I}{\lambda y.\mathsf{i}_Q(\mathsf{o}_P(f(y))) : \mathbf{M}_k \to \{x : \mathbf{entity} \mid Q(x)\}}\,\Pi I, 1$$

We next prove that $\lambda y.\mathsf{i}_Q(\mathsf{o}_P(f(y)))$ is a bijection.

To prove $\mathbf{injection}\,(\lambda y.\mathsf{i}_Q(\mathsf{o}_P(f(y))))$, we assume $y : \mathbf{M}_k$, $y' : \mathbf{M}_k$, and $u : \mathsf{i}_Q(\mathsf{o}_P(f(y))) =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(y')))$ and show that $y =_{\mathbf{M}_k} y'$ holds.

From $f : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\}$ and the assumptions y and y', we

have:

$$f(y) : \{x : \mathbf{entity} \mid P(x)\}$$
$$f(y') : \{x : \mathbf{entity} \mid P(x)\}$$

With $f(y)$, $f(y')$, and $u : \mathsf{i}_Q(\mathsf{o}_P(f(y))) =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(y')))$, it follows that $f(y) =_{\{x:\mathbf{entity}|P(x)\}} f(y')$ holds by Theorem 28 (see page 210). Therefore, as f is an injection, we conclude that $y = y'$.

To prove **surjection** $(\lambda y.\mathsf{i}_Q(\mathsf{o}_P(f(y))))$, we assume $z : \{x : \mathbf{entity} \mid Q(x)\}$ and construct a proof of $\begin{bmatrix} y : \mathbf{M}_k \\ z =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(y))) \end{bmatrix}$.

By the derivation,

$$\dfrac{\dfrac{\dfrac{z' : \{x : \mathbf{entity} \mid Q(x)\}}{\mathsf{o}_Q(z') : \mathbf{entity}}\{|\}E \quad \dfrac{t_2 : (x : \mathbf{entity}) \to Q(x) \to P(x) \qquad \mathsf{o}_Q(z') : \mathbf{entity}}{t_2(\mathsf{o}_Q(z')) : Q(\mathsf{o}_Q(z')) \to P(\mathsf{o}_Q(z'))}\Pi E \quad \dfrac{?(z') : Q(\mathsf{o}_Q(z'))}{?(z') : Q(\mathsf{o}_Q(z'))}\{|\}E_?}{(t_2(\mathsf{o}_Q(z')))(?(z')) : P(\mathsf{o}_Q(z'))}\Pi E}{\{|\}I}$$

$$\dfrac{\mathsf{i}_P(\mathsf{o}_Q(z'))f(y) : \{x : \mathbf{entity} \mid P(x)\} \qquad \dfrac{\dfrac{b_f : \mathbf{bijection}(f)}{\pi_2(b_f) : \mathbf{surjection}(f)}\Sigma E}{\pi_2(b_f) : (z : \{x : \mathbf{entity} \mid P(x)\}) \to \begin{bmatrix} y : \mathbf{M}_k \\ z =_{\{x:\mathbf{entity}|P(x)\}} f(y) \end{bmatrix}}\,def}{(\pi_2 b_f)(\mathsf{i}_P(\mathsf{o}_Q(z'))f(y)) : \begin{bmatrix} y : \mathbf{M}_k \\ \mathsf{i}_P(\mathsf{o}_Q(z')) =_{\{x:\mathbf{entity}|P(x)\}} f(y) \end{bmatrix}}\Pi E,$$

we obtain

$$(\pi_2 b_f)(\mathsf{i}_P(\mathsf{o}_Q(z'))f(y)) : \begin{bmatrix} y : \mathbf{M}_k \\ \mathsf{i}_P(\mathsf{o}_Q(z')) =_{\{x:\mathbf{entity}|P(x)\}} f(y) \end{bmatrix}. \tag{5.39}$$

We write this w. By applying ΣE, we get

$$\pi_1 w : \mathbf{M}_k \tag{5.40}$$
$$\pi_2 w : \mathsf{i}_P(\mathsf{o}_Q(z')) =_{\{x:\mathbf{entity}|P(x)\}} f(\pi_1 w) \tag{5.41}$$

Now, the following holds.

$$\mathsf{refl} : \mathsf{i}_Q(\mathsf{o}_P(\mathsf{i}_P(\mathsf{o}_Q(z')))) =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(\mathsf{i}_P(\mathsf{o}_Q(z')))) \tag{5.42}$$

With (5.41), we get the following formula by substitution.

$$\mathsf{refl} : \mathsf{i}_Q(\mathsf{o}_P(\mathsf{i}_P(\mathsf{o}_Q(z')))) =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(\pi_1 w))) \tag{5.43}$$

As $\mathsf{i}_Q(\mathsf{o}_P(\mathsf{i}_P(\mathsf{o}_Q(z')))) \equiv_\beta z'$, we get

$$\mathsf{refl} : z' =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(\pi_1 w))). \tag{5.44}$$

Therefore, by applying ΣI with (5.40) and (5.44), we can construct the proof:

$$\langle \pi_1(w), \mathsf{refl} \rangle : \left[\begin{array}{l} y : \mathbf{M}_k \\ z =_{\{x:\mathbf{entity}|Q(x)\}} \mathsf{i}_Q(\mathsf{o}_P(f(y))) \end{array} \right] \tag{5.45}$$

End of proof.

5.3.4 Theorem: subset types with disjoint specification

Theorem 5

For any $P,\ Q : \mathbf{entity} \to \mathbf{type}$ and $k : \mathbf{Nat}$,

$$\begin{array}{l} \Gamma,\ t : (x : \mathbf{entity}) \to P(x) \to Q(x) \to \bot, \\ f : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x) \uplus Q(x)\},\ b_f : \mathbf{bijection}(f) \end{array} \vdash \left[\begin{array}{l} f^* : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\} \\ \mathbf{bijection}(f^*) \end{array} \right] \ \mathit{true}$$

Proof. We first construct a function that has type $\mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}$.

Assume $y : \mathbf{M}_k$. With f, we obtain $f(y) : \{x : \mathbf{entity} \mid P(x) \uplus Q(x)\}$, from which we get the following proofs by the subset elimination rules (we abbreviate $\lambda x.P(x) \uplus Q(x)$ as PQ.

$$\mathsf{o}_{PQ}(f(y)) : \mathbf{entity} \tag{5.46}$$

$$?(f(y)) : P(\mathsf{o}_{PQ}(f(y))) \uplus Q(\mathsf{o}_{PQ}(f(y))) \tag{5.47}$$

By (5.47), proof can be divided into two cases, (i) and (ii):
(i) Assume $z : P(\mathsf{o}_{PQ}(f(y)))$. We get the following by $\{|\}I$ together with (5.46).

$$\mathsf{i}_P(\mathsf{o}_{PQ}(f(y))) : \{x : \mathbf{entity} \mid P(x)\} \tag{5.48}$$

By $\uplus I$,

$$\mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(y)))) : \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\} \tag{5.49}$$

(ii) Assume $z : Q(\mathsf{o}_{PQ}(f(y)))$. We get the following by $\{|\}I$ together with (5.46).

$$\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y))) : \{x : \mathbf{entity} \mid Q(x)\} \tag{5.50}$$

By $\uplus I$,

$$\mathsf{inr}(\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y)))) : \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\} \tag{5.51}$$

From (i) and (ii), by $\uplus E$, we obtain

$$\begin{aligned}&\mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(y)))), \lambda z.\mathsf{inr}(\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y)))))\\ &\qquad : \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}\end{aligned} \tag{5.52}$$

Thus, by applying ΠI to $y : \mathbf{M}_k$, we obtain the function

$$\begin{aligned}&\lambda y.\mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(y)))), \lambda z.\mathsf{inr}(\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y)))))\\ &\qquad : \mathbf{M}_k \to \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}\,.\end{aligned} \tag{5.53}$$

We abbreviate the function as $\lambda y.\mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y))$.

Next, we will prove that (5.53) is an injection. We assume $y : \mathbf{M}_k$ and $y' : \mathbf{M}_k$, and show that the following holds.

$$\begin{aligned}\mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ ?(f(y'))\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))\\ \rightarrow y =_{\mathbf{M}_k} y'\end{aligned} \tag{5.54}$$

By $?(f(y))$ and $?(f(y'))$, which have both the type $\{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}$, the proof is divided into four cases.

1. $\mathsf{case}\ \mathsf{inl}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inl}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))$
   $\rightarrow y =_{\mathbf{M}_k} y'$

2. $\mathsf{case}\ \mathsf{inl}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inr}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))$
   $\rightarrow y =_{\mathbf{M}_k} y'$

3. $\mathsf{case}\ \mathsf{inr}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inl}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))$
   $\rightarrow y =_{\mathbf{M}_k} y'$

4. $\mathsf{case}\ \mathsf{inr}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inr}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))$
   $\rightarrow y =_{\mathbf{M}_k} y'$

In cases 2 and 3, the equation in the antecedent clause leads to contradiction as their canonical form is different (see Theorem 25 on page 204). Thus, we can apply $\perp E$ to derive the consequence.

In case 1,

$$\begin{aligned}&\mathsf{case}\ \mathsf{inl}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inl}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y'))\\ \equiv\ &L(y) = L(y')\\ \equiv\ &\mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(y)))) = \mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(y'))))\end{aligned} \tag{5.55}$$

As $\lambda x.\mathsf{inl}(x)$ is an injection (see Lemma 20 on page 200),

$$\mathsf{i}_P(\mathsf{o}_{PQ}(f(y))) = \mathsf{i}_P(\mathsf{o}_{PQ}(f(y'))) \tag{5.56}$$

By Theorem 28, we can eliminate i and o, so as to get $f(y) = f(y')$. As f is an injection, we can conclude that $y = y'$ holds.

The proof of the last case, 4, is similar to case 1.

$$\begin{aligned} & \mathsf{case}\ \mathsf{inr}(t)\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) = \mathsf{case}\ \mathsf{inr}(t')\ \mathsf{of}\ (\lambda z.L(y'), \lambda z.R(y')) \\ \equiv\ & R(y) = R(y') \\ \equiv\ & \mathsf{inr}(\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y)))) = \mathsf{inr}(\mathsf{i}_Q(\mathsf{o}_{PQ}(f(y')))) \end{aligned} \tag{5.57}$$

Again, because $\lambda x.\mathsf{inr}(x)$ is an injection (Lemma 21) and we can eliminate i and o (Theorem 28), we obtain $f(y) = f(y')$. Thus, $y = y'$.

By combining cases 1 to 4 above and applying $\uplus E$ several times, we can conclude that (5.54) holds.

Lastly, we will prove that (5.53) is also a surjection. We assume $z : \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}$ and prove $\left[\begin{array}{l} y : \mathbf{M}_k \\ z = \mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) \end{array}\right]$.

By z, the proof can be divided into two cases, either it is $\mathsf{inl}(z')$ or $\mathsf{inr}(z')$ for some z'.

(I) Assume $z' : \{x : \mathbf{entity} \mid P(x)\}$. We will give a proof term of

$$\left[\begin{array}{l} y : \mathbf{M}_k \\ \mathsf{inl}(z') = \mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) \end{array}\right]. \tag{5.58}$$

First, by the following derivation,

$$\dfrac{\dfrac{b_f : \mathbf{bijection}(f)}{(\pi_2 b_f) : (z : \{x : \mathbf{entity} \mid P(x) \uplus Q(x)\}) \to \left[\begin{array}{l} y : \mathbf{M}_k \\ z = f(y) \end{array}\right]}\,\Sigma E \quad \dfrac{\dfrac{\dfrac{z' : \{x : \mathbf{entity} \mid P(x)\}}{?(z') : P(\mathsf{o}_P(z'))}\,\{|\}E_?}{\mathsf{inl}(?(z')) : P(\mathsf{o}_P(z')) \uplus Q(\mathsf{o}_P(z'))}\,\uplus I \quad \dfrac{z' : \{x : \mathbf{entity} \mid P(x)\}}{\mathsf{o}_P(z') : \mathbf{entity}}\,\{|\}E}{\mathsf{i}_{PQ}(\mathsf{o}_P(z')) : \{x : \mathbf{entity} \mid P(x) \uplus Q(x)\}}\,\{|\}I}{(\pi_2 b_f)(\mathsf{i}_{PQ}(\mathsf{o}_P(z'))) : \left[\begin{array}{l} y : \mathbf{M}_k \\ \mathsf{i}_{PQ}(\mathsf{o}_P(z')) =_{\{x:\mathbf{entity} \mid P(x) \uplus Q(x)\}} f(y) \end{array}\right]}\,\Pi E ,$$

we construct

$$(\pi_2 b_f)(\mathsf{i}_{PQ}(\mathsf{o}_P(z'))) : \left[\begin{array}{l} y : \mathbf{M}_k \\ \mathsf{i}_{PQ}(\mathsf{o}_P(z')) =_{\{x:\mathbf{entity}|P(x)\uplus Q(x)\}} f(y) \end{array}\right]. \tag{5.59}$$

We abbreviate $(\pi_2 b_f)(\mathsf{i}_{PQ}(\mathsf{o}_P(z')))$ as w. From (5.59), we obtain

$$\pi_1 w : \mathbf{M}_k \tag{5.60}$$

$$\pi_2 w : \mathsf{i}_{PQ}(\mathsf{o}_P(z')) =_{\{x:\mathbf{entity}|P(x)\uplus Q(x)\}} f(\pi_1 w) \tag{5.61}$$

Now, let us prove

$$\begin{array}{l} ?(f(\pi_1 w)) = ?(f(\pi_1 w)) \\ \qquad \to \mathsf{inl}(z') = \mathsf{case}\ ?(f(\pi_1 w))\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)). \end{array} \tag{5.62}$$

Once it is proved, by applying ΠE with (5.62) and $\mathsf{refl} : ?(f(\pi_1 w)) = ?(f(\pi_1 w))$, we obtain

$$\mathsf{inl}(z') = \mathsf{case}\ ?(f(\pi_1 w))\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)), \tag{5.63}$$

from which, by applying ΣI together with (5.60), we obtain our goal, (5.58).

The proof of (5.62) is divided into two cases, with respect to $?(f(\pi_1 w)) : P(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_P(z')))) \uplus Q(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_P(z'))))$.

(I-i) Assume $v : P(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_P(z'))))$. We will prove

$$\begin{array}{l} ?(f(\pi_1 w)) = \mathsf{inl}(v) \\ \qquad \to \mathsf{inl}(z') = \mathsf{case}\ ?(f(\pi_1 w))\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)). \end{array}$$

By assuming $?(f(\pi_1 w)) = \mathsf{inl}(v)$, the following equation holds. Here,

(5.68) is obtained by the substitution with (5.61).

$$\begin{aligned}
& \mathsf{case}\ ?(f(\pi_1 w))\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)) && (5.64)\\
\equiv\ & \mathsf{case}\ \mathsf{inl}(v)\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)) && (5.65)\\
\equiv\ & L(\pi_1 w) && (5.66)\\
\equiv\ & \mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(f(\pi_1 w)))) && (5.67)\\
=\ & \mathsf{inl}(\mathsf{i}_P(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_p(z'))))) && (5.68)\\
\equiv\ & \mathsf{inl}(z') && (5.69)
\end{aligned}$$

(I-ii) Assume $v : Q(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_P(z'))))$. We will prove

$$\begin{aligned}
& ?(f(\pi_1 w)) = \mathsf{inr}(v)\\
& \quad \to \mathsf{inl}(z') = \mathsf{case}\ ?(f(\pi_1 w))\ \mathsf{of}\ (\lambda z.L(\pi_1 w), \lambda z.R(\pi_1 w)).
\end{aligned} \tag{5.70}$$

As $Q(\mathsf{o}_{PQ}(\mathsf{i}_{PQ}(\mathsf{o}_P(z')))) \equiv Q(\mathsf{o}_P(z'))$, we have $v : Q(\mathsf{o}_P(z'))$. On the other hand, as we have assumed $z' : \{x : \mathbf{entity} \mid P(x)\}$, we also have $?(z')P(\mathsf{o}_P(z'))$. Thus, from the assumption $t : (x : \mathbf{entity}) \to P(x) \to Q(x) \to \bot$, we obtain a contradiction. Therefore, by $\bot E$, we can deduce (5.70).

By combining (I-i) and (I-ii), (5.62) is proved.

(II) The proof of the other side can be given analogously by assuming $z' : \{x : \mathbf{entity} \mid Q(x)\}$.

From (I) and (II) above, by $\uplus E$, we obtain a proof of

$$\begin{bmatrix} y : \mathbf{M}_k \\ z = \mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) \end{bmatrix}.$$

Therefore, we conclude that a proof exists for

$$(z : \{x : \mathbf{entity} \mid P(x)\} \uplus \{x : \mathbf{entity} \mid Q(x)\}) \to \begin{bmatrix} y : \mathbf{M}_k \\ z = \mathsf{case}\ ?(f(y))\ \mathsf{of}\ (\lambda z.L(y), \lambda z.R(y)) \end{bmatrix}.$$

End of proof.

5.3.5 Lemma: segmentation of a type

We are to prove the theorem that says that if some type is finite, then its part is also finite.

Theorem

Given A : **type**, B : **type**, and k : **Nat**, if there exists a function f : $\mathbf{M}_k \rightarrow A \uplus B$ such that **bijection** (f) is true, there exists a function $g : \mathbf{M}_m \rightarrow A$ for some $m \leq k$ such that **bijection** (g) is true.

Here, g is a function whose co-domain is A, which is a part of the co-domain of f. We want to construct such a function from f. However, it is impossible unless we know, for each of the elements in $\mathbf{M}_k$, either it goes to A or B. Therefore, here we consider an additional function, h. The function h is designed as follows: for each element in $\mathbf{M}_k$, if f maps it to an element of A, h maps it to $\mathbf{M}_m$; otherwise, h maps it to $\mathbf{M}_n$ (Lemma 8).

To prove this inductively, we use $\mathbf{M}_{k'}$, where $k' \leq k$. There always exists an injection from $\mathbf{M}_{k'}$ to $\mathbf{M}_k$ that can connect all the element in $\mathbf{M}_{k'}$ with the first k' elements in $\mathbf{M}_k$. We formulate this as Lemma 6.[3]

[3] Hereafter, we use the abbreaviated notation $(x, x' : A) \rightarrow B$ for $(x : A) \rightarrow (x' : A) \rightarrow B$ if there is no confusion.

Preparation

Lemma 6: $\Gamma \vdash (k' : \mathbf{Nat}) \to (n : \mathbf{Nat}) \to \left[\begin{array}{l} f : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n} \\ \mathbf{injection}\,(f) \end{array}\right]$ *true*

Proof. Given $k' : \mathbf{Nat}$, we prove the consequence by induction on $n : \mathbf{Nat}$.

(1) First we give the proof for the case $n = 0$. We can prove the existence of a function of $\mathbf{M}_{k'} \to \mathbf{M}_{k'}$ by the identity function:

$$id_{\mathbf{M}_{k'}} : \mathbf{M}_{k'} \to \mathbf{M}_{k'}.$$

Now, for any $x, y : \mathbf{M}_{k'}$, if $id_{\mathbf{M}_{k'}}(x) = id_{\mathbf{M}_{k'}}(y)$ then $x = y$. Hence, we have the following proof, which we write $I_{id_{\mathbf{M}_{k'}}}$.

$$I_{id_{\mathbf{M}_{k'}}} : \mathbf{injection}\left(id_{\mathbf{M}_{k'}}\right)$$

Thus, by ΣI, we get

$$(id_{\mathbf{M}_{k'}}, I_{id_{\mathbf{M}_{k'}}}) : \left[\begin{array}{l} f : \mathbf{M}_{k'} \to \mathbf{M}_{k'+0} \\ \mathbf{injection}\,(f) \end{array}\right]. \tag{5.71}$$

(2) Assume that we have the proof $u : (f' : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n'}) \times \mathbf{injection}\,(f')$ for $n' : \mathbf{Nat}$. We will give the proof for $\mathsf{suc}(n')$. For any $x : \mathbf{M}_{k'}$,

$$(\pi_1 u)(x) : \mathbf{M}_{k'+n'}.$$

By $\uplus I_L$,

$$\mathsf{inl}((\pi_1 u)(x)) : \mathbf{M}_{k'+n'} \uplus T.$$

Hence, by ΠI,

$$\lambda x.\mathsf{inl}((\pi_1 u)(x)) : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n'} \uplus T.$$

As $\mathbf{M}_{k'+n'} \uplus T \equiv \mathbf{M}_{\mathsf{suc}(k'+n')} \equiv \mathbf{M}_{k'+\mathsf{suc}(n')}$,

$$\lambda x.\mathsf{inl}((\pi_1 u)(x)) : \mathbf{M}_{k'} \to \mathbf{M}_{k'+\mathsf{suc}(n')}. \tag{5.72}$$

From IH, we have $\pi_2 u$: **injection** $(\pi_1 u)$. Also, **injection** $(\lambda x.\mathsf{inl}(x))$ holds for $\lambda x.\mathsf{inl}(x)$ (see Lemma 20 on page 200). As the composition of two injections is also an injection (see Lemma 22 on page 201), we can obtain a proof of **injection** $(\lambda x.\mathsf{inl}(\pi_1 u(x)))$. We write it $I_{\lambda x.\mathsf{inl}((\pi_1 u)(x))}$ as below.

$$I_{\lambda x.\mathsf{inl}((\pi_1 u)(x))} : \mathbf{injection}\,(\lambda x.\mathsf{inl}((\pi_1 u)(x))) \tag{5.73}$$

By applying ΣI to (5.72) and (5.73),

$$(\lambda x.\mathsf{inl}((\pi_1 u)(x)), I_{\lambda x.\mathsf{inl}((\pi_1 u)(x))}) : \left[\begin{array}{l} f'' : \mathbf{M}_{k'} \to \mathbf{M}_{k'+\mathsf{suc}(n')} \\ \mathbf{injection}\,(f'') \end{array}\right].$$

Hence, by ΠI, we obtain

$$\begin{array}{l} \lambda n'.\lambda u.(\lambda x.\mathsf{inl}((\pi_1 u)(x)), I_{\lambda x.\mathsf{inl}((\pi_1 u)(x))}) : \\ \quad (n' : \mathbf{Nat}) \to \left[\begin{array}{l} f' : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n'} \\ \mathbf{injection}\,(f') \end{array}\right] \to \left[\begin{array}{l} f'' : \mathbf{M}_{k'} \to \mathbf{M}_{k'+\mathsf{suc}(n')} \\ \mathbf{injection}\,(f'') \end{array}\right]. \end{array} \tag{5.74}$$

Combining (1) with (2), that is, by applying $\mathbf{Nat}E$ to (5.71) and (5.74), we get

$$\mathsf{natrec}(n, (id_{\mathbf{M}_{k'}}, I_{id_{\mathbf{M}_{k'}}}), \lambda n'.\lambda u.(\lambda x.\mathsf{inl}((\pi u)(x)), I_{\lambda x.\mathsf{inl}((\pi u)(x))})) : \left[\begin{array}{l} f : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n} \\ \mathbf{injection}\,(f) \end{array}\right]$$

for any n and the given k'. Therefore,

$$\lambda k'.\lambda n.\mathsf{natrec}(n, (id_{\mathbf{M}_{k'}}, I_{id_{\mathbf{M}_{k'}}}), \lambda n'.\lambda u.(\lambda x.\mathsf{inl}((\pi u)(x)), I_{\lambda x.\mathsf{inl}((\pi u)(x))})) : \left[\begin{array}{l} f : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n} \\ \mathbf{injection}\,(f) \end{array}\right].$$

End of proof.

The following judgement immediately follows from Lemma 6.[4]

[4] When we apply the same inference rules successively in an obvious way, we will omit

Corollary 7:

$$\Gamma \vdash (k, k' : \mathbf{Nat}) \to (u : k' \leq k) \to \left[\begin{array}{l} i : \mathbf{M}_{k'} \to \mathbf{M}_k \\ \mathbf{injection}\,(i) \end{array}\right] \; true$$

Proof. The following derivation shows the construction of the proof, where **p** stands for the proof of Lemma 6.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{k' : \mathbf{Nat}}\,1 \quad \dfrac{\dfrac{}{u : \left[\begin{array}{l} n : \mathbf{Nat} \\ k = k' + n \end{array}\right]}\,1}{\pi_1(u) : \mathbf{Nat}}\,\Sigma E_L}{\mathbf{p} : (k' : \mathbf{Nat}) \to (n : \mathbf{Nat}) \to \left[\begin{array}{l} i : \mathbf{M}_{k'} \to \mathbf{M}_{k'+n} \\ \mathbf{injection}\,(i) \end{array}\right]}\,(\text{Lm6}) \quad \dfrac{}{u : \left[\begin{array}{l} n : \mathbf{Nat} \\ k = k' + n \end{array}\right]}\,1}{\mathbf{p}(k')(\pi_1(u)) : \left[\begin{array}{l} i : \mathbf{M}_{k'} \to \mathbf{M}_{k'+\pi_1(u)} \\ \mathbf{injection}\,(i) \end{array}\right]}\,\Pi E^* \quad \dfrac{}{\pi_2(u) : k = k' + \pi_1(u)}\,\Sigma E_R}{\mathbf{p}(k')(\pi_1(u)) : \left[\begin{array}{l} i : \mathbf{M}_{k'} \to \mathbf{M}_k \\ \mathbf{injection}\,(i) \end{array}\right]}\,conv \quad \dfrac{}{k : \mathbf{Nat}}\,1}{\lambda k.\lambda k'.\lambda u.\mathbf{p}(k')(\pi_1(u)) : (k, k' : \mathbf{Nat}) \to (u : k' \leq k) \to \left[\begin{array}{l} i : \mathbf{M}_{k'} \to \mathbf{M}_k \\ \mathbf{injection}\,(i) \end{array}\right]}\,\Pi I, 1^*$$

End of proof.

We have proved that we can construct an injection from $\mathbf{M}_{k'}$ to $\mathbf{M}_k$ whenever $k' \leq k$ holds. For simplicity of notation, we let $i_{k' \leq k}$ stand for such an injection of type $\mathbf{M}_{k'} \to \mathbf{M}_k$.

Proof of Lemma 8

We are to construct a function $h : \mathbf{M}_{k'} \to \mathbf{M}_k$ that we mentioned in the beginning of this section. The function h maps every element x in $\mathbf{M}_{k'}$ to $\mathbf{M}_m$ if f maps $i_{k' \leq k}(x)$ to an element of A. Otherwise, h maps it to $\mathbf{M}_n$. The existence of such h is formulated as Lemma 8.

the derivation steps by indicating this by the superscript $*$ next to the inference rules' label.

Lemma 8:

$$\Gamma,\ A : \mathbf{type},\ B : \mathbf{type} \vdash$$

$$(k, k' : \mathbf{Nat}) \to (u : k' \le k) \to (f : \mathbf{M}_k \to A \uplus B) \to \mathbf{bijection}\,(f) \to$$

$$\left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} k' = m + n \\ \left[\begin{array}{l} h : \mathbf{M}_{k'} \to \mathbf{M}_m \uplus \mathbf{M}_n \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : \mathbf{M}_{k'}) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{k' \le k}(x)) =_{A \uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_m \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{k' \le k}(x)) =_{A \uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_n \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \quad \mathit{true}$$

Proof. Let k : **Nat** be an arbitrary number. Proof is given by induction on k' : **Nat**.

(1) First, we prove the following formula for the case $k' = 0$.

$$(k : \mathbf{Nat}) \to (u_0 : 0 \le k) \to (f : M(k) \to A \uplus B) \to \mathbf{bijection}\,(f) \to$$

$$\left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} 0 = m + n \\ \left[\begin{array}{l} h : M(0) \to M(m) \uplus M(n) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0 \le k}(x)) =_{A \uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x) =_{M(m) \uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0 \le k}(x)) =_{A \uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x) =_{M(m) \uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] .$$

Assume k : **Nat**, u_o : $0 \le k$, f : $M(k) \to A \uplus B$, **bijection**$\,(f)$. The following derivation holds, where q_1 and q_2 are proved below in $\mathcal{D}'_1$ and $\mathcal{D}'_2$.

$$
\dfrac{
\dfrac{\mathcal{D}_1'}{q_1 : 0 = 0+0}
\qquad
\dfrac{\mathcal{D}_2'}{q_2 : \left[\begin{array}{l} h : M(0) \to M(0) \uplus M(0) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]}
}{
\dfrac{
\dfrac{}{0 : \mathbf{Nat}}\,\mathbf{Nat}I_0
\qquad
\langle q_1, q_2\rangle : \left[\begin{array}{l} 0 = 0+0 \\ \left[\begin{array}{l} h : M(0) \to M(0) \uplus M(0) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]
}{
\dfrac{
\dfrac{}{0 : \mathbf{Nat}}\,\mathbf{Nat}I_0
\qquad
\langle 0, \langle q_1, q_2\rangle\rangle : \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} 0 = 0+n \\ \left[\begin{array}{l} h : M(0) \to M(0) \uplus M(n) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f\left((\pi_1(\mathbf{i}(k)(0)(u_0)))(x)\right) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f\left((\pi_1(\mathbf{i}(k)(0)(u_0)))(x)\right) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x) =_{M(0)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]
}{
\langle 0, \langle 0, \langle q_1, q_2\rangle\rangle\rangle : \left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} 0 = m+n \\ \left[\begin{array}{l} h : M(0) \to M(m) \uplus M(n) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f\left((\pi_1(\mathbf{i}(k)(0)(u_0)))(x)\right) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f\left((\pi_1(\mathbf{i}(k)(0)(u_0)))(x)\right) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]
}\,\Sigma I
}\,\Sigma I
}\,\times I
\tag{5.75}
$$

$\mathcal{D}_1'$:

$$
\dfrac{
\dfrac{}{\mathbf{Nat} : \mathbf{type}}
\qquad
\dfrac{
\dfrac{}{\mathbf{Nat} : \mathbf{type}}
\qquad
\dfrac{\dfrac{}{0 : \mathbf{Nat}}\,\mathbf{Nat}I_0 \qquad \dfrac{}{0 : \mathbf{Nat}}\,\mathbf{Nat}I_0}{0+0 : \mathbf{Nat}}
\qquad
\dfrac{}{0 : \mathbf{Nat}}\,\mathbf{Nat}I_0
\qquad
\dfrac{}{0+0 \equiv_\beta 0}\,(\text{Th17})
}{\mathsf{refl}_{\mathbf{Nat}}(0) : 0+0 = 0}\,(\text{Th16})
}{\mathbf{p}_{13}(\mathsf{refl}_{\mathbf{Nat}}(0)) : 0 = 0+0}\,{=}\text{Sym}
$$

Thus, q_1 in (5.75) is $\mathbf{p}_{13}(\mathsf{refl}_{\mathbf{Nat}}(0))$.

$\mathcal{D}_2'$:

First, as $M(0) \overset{def}{\equiv} \bot$, we obtain $\lambda z.\mathsf{case}_z() : M(0) \to M(0) \uplus M(0)$ from the following delivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{z : \bot}^{\,1}}{\mathsf{case}_z() : \bot \uplus \bot}\,\bot E}{\lambda z.\mathsf{case}_z() : \bot \to \bot \uplus \bot}\,\Pi I, 1}}{} \qquad \dfrac{\dfrac{\overline{z : \bot}^{\,2} \quad \dfrac{\overline{\vdots}}{\bot \uplus \bot : \mathbf{type}}}{\bot \uplus \bot : \mathbf{type}}\,wk}{\lambda z.\bot \uplus \bot : \bot \to \mathbf{type}}\,\Pi I, 2 \tag{5.76}$$

Second, we show that $\lambda z.\mathsf{case}_z() : M(0) \to M(0) \uplus M(0)$ is a bijection. By definition,

$$\mathbf{bijection}\,(\lambda z.\mathsf{case}_z()) \equiv \left[\begin{array}{l} \mathbf{injection}\,(\lambda z.\mathsf{case}_z()) \\ \mathbf{surjection}\,(\lambda z.\mathsf{case}_z()) \end{array}\right].$$

As $\mathbf{injection}\,(\lambda z.\mathsf{case}_z()) \equiv (y : \bot) \to (y' : \bot) \to (\lambda z.\mathsf{case}_z())(y) = (\lambda z.\mathsf{case}_z())(y') \to y = y'$, the proof is given by the following delivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{y : \bot}^{\,1} \quad \dfrac{\dfrac{\overline{z' : \bot}^{\,2}}{\vdots}}{(y' : \bot) \to (\lambda z.\mathsf{case}_z())(z') = (\lambda z.\mathsf{case}_z())(y') \to z' = y' : \mathbf{type}}}{\lambda z'.(y' : \bot) \to (\lambda z.\mathsf{case}_z())(z') = (\lambda z.\mathsf{case}_z())(y') \to z' = y' : \bot \to \mathbf{type}}\,\Pi I, 2}{\mathsf{case}_y() : (y' : \bot) \to (\lambda z.\mathsf{case}_z())(y) = (\lambda z.\mathsf{case}_z())(y') \to y = y'}\,\bot E}{\lambda y.\mathsf{case}_y() : (y : \bot) \to (y' : \bot) \to (\lambda z.\mathsf{case}_z())(y) = (\lambda z.\mathsf{case}_z())(y') \to y = y'}\,\Pi I, 1$$

On the other hand, the proof is given by the following delivation as $\mathbf{surjection}\,(\lambda z.\mathsf{case}_z()) \equiv (z' : \bot \uplus \bot) \to \left[\begin{array}{l} y : \bot \\ z' = (\lambda z.\mathsf{case}_z())(y) \end{array}\right]$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{z' : \bot}\,3}{\vdots}}{\left[\begin{array}{l} y : \bot \\ \mathsf{inl}(z') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] : \mathbf{type}}\,\Pi I, 3}{\lambda z'.\left[\begin{array}{l} y : \bot \\ \mathsf{inl}(z') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] : \bot \to \mathbf{type}} \quad \dfrac{\dfrac{\dfrac{}{z'' : \bot}\,2}{\mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inl}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\bot E}{\dfrac{\dfrac{}{z' : \bot \uplus \bot}\,1 \quad \mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inl}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] \quad \mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inr}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}{\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z''.\mathsf{case}_{z''}(), \lambda z''.\mathsf{case}_{z''}()) : \left[\begin{array}{l} y : \bot \\ z' = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\uplus E, 2}\,\Pi I, 1}{\lambda z'.\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z''.\mathsf{case}_{z''}(), \lambda z''.\mathsf{case}_{z''}()) : (z' : \bot \uplus \bot) \to \left[\begin{array}{l} y : \bot \\ z' = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\Pi I, 1}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{z' : \bot}\,4}{\vdots}}{\left[\begin{array}{l} y : \bot \\ \mathsf{inr}(z') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] : \mathbf{type}}\,\Pi I, 4}{\lambda z'.\left[\begin{array}{l} y : \bot \\ \mathsf{inr}(z') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] : \bot \to \mathbf{type}} \quad \dfrac{\dfrac{\dfrac{}{z'' : \bot}\,2}{\mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inr}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\bot E}{\dfrac{\dfrac{}{z' : \bot \uplus \bot}\,1 \quad \mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inl}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right] \quad \mathsf{case}_{z''}() : \left[\begin{array}{l} y : \bot \\ \mathsf{inr}(z'') = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}{\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z''.\mathsf{case}_{z''}(), \lambda z''.\mathsf{case}_{z''}()) : \left[\begin{array}{l} y : \bot \\ z' = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\uplus E, 2}\,\Pi I, 1}{\lambda z'.\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z''.\mathsf{case}_{z''}(), \lambda z''.\mathsf{case}_{z''}()) : (z' : \bot \uplus \bot) \to \left[\begin{array}{l} y : \bot \\ z' = (\lambda z.\mathsf{case}_z()))(y) \end{array}\right]}\,\Pi I, 1$$

Therefore, by ΣI, we obtain

$$\langle \lambda y.\mathsf{case}_y(), \lambda z'.\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z'.\mathsf{case}_{z'}(), \lambda z'.\mathsf{case}_{z'}())\rangle : \mathbf{bijection}\,(\lambda z.\mathsf{case}_z()). \tag{5.77}$$

Third, the following delivation holds.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : \bot}\,2}{\vdots}}{\left[\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \quad \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right]\right] : \mathbf{type}}\,\Pi I, 2}{\lambda x.\left[\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \quad \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right]\right] : \bot \to \mathbf{type}}\,\bot E}{\mathsf{case}_x() : \left[\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \quad \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right]\right]}\,\Pi I, 1}{\lambda x.\mathsf{case}_x() : (x : \bot) \to \left[\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \quad \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ (\lambda z.\mathsf{case}_z())(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right]\right]}$$

(5.78)

Thus, by applying ΣI with (5.76), (5.77), and (5.78) in order, we obtain:

$$\langle \lambda z.\mathsf{case}_z(), \langle\langle \lambda y.\mathsf{case}_y(), \lambda z'.\mathsf{case}\ z'\ \mathsf{of}\ (\lambda z'.\mathsf{case}_{z'}(), \lambda z'.\mathsf{case}_{z'}())\rangle, \lambda x.\mathsf{case}_x()\rangle\rangle :$$
$$\left[\begin{array}{l} h : M(0) \to M(0) \uplus M(0) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(0)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{0\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(0) \\ h(x) =_{M(0)\uplus M(0)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right].$$

This is what q_2 stands for in (5.75).

(2) Assume that the formula holds for $k' = d$, i.e., we have the following proof term $\mathcal{IH}$.

$$\mathcal{IH} : \begin{array}{l} (k : \mathbf{Nat}) \to (u : d \le k) \to (f : M(k) \to A \uplus B) \to \mathbf{bijection}\,(f) \to \\ \left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} d = m + n \\ \left[\begin{array}{l} h : M(d) \to M(m) \uplus M(n) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : M(d)) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{d\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{d\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right].$$

We will prove it for $k' = \mathsf{suc}(d)$, i.e., the proof of the following proposition exists.

$$\begin{array}{l} (k : \mathbf{Nat}) \to (u' : \mathsf{suc}(d) \le k) \to (f : M(k) \to A \uplus B) \to \mathbf{bijection}\,(f) \to \\ \left[\begin{array}{l} m' : \mathbf{Nat} \\ \left[\begin{array}{l} n' : \mathbf{Nat} \\ \left[\begin{array}{l} \mathsf{suc}(d) = m' + n' \\ \left[\begin{array}{l} h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \left[\begin{array}{l} \mathbf{bijection}\,(h') \\ (x : M(\mathsf{suc}(d))) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}$$

(5.79)

Assume $k : \mathbf{Nat}$, $u' : \mathsf{suc}(d) \le k$, $f : M(k) \to A \uplus B$, and $q : \mathbf{bijection}\,(f)$.

From $\mathcal{IH}$ and these assumptions, by applying ΠE several times, we obtain

$$\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q):$$
$$\left[\begin{array}{l} m:\mathbf{Nat} \\ \left[\begin{array}{l} n:\mathbf{Nat} \\ \left[\begin{array}{l} d = m+n \\ \left[\begin{array}{l} h: M(d) \to M(m) \uplus M(n) \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x:M(d)) \to \left[\begin{array}{l} \left(\begin{array}{l} v_1: \left[\begin{array}{l} z:A \\ f(i_{d\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right] \end{array}\right) \to \left[\begin{array}{l} w:M(m) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(\begin{array}{l} v_2: \left[\begin{array}{l} z:B \\ f(i_{d\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right] \end{array}\right) \to \left[\begin{array}{l} w:M(n) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right],$$

where $\mathbf{p}_{19}(u')$ is a proof of $d \le k$ obtained from $u' : \mathsf{suc}(d) \le k$ and Theorem 19. From the proof above, we obtain each of the following proof terms by applying ΣE several times. We write the letters shown in the left hand side for the proof terms in the right hand side.

$$\begin{array}{rcl} m & \equiv & \pi_1(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : \mathbf{Nat} \\ n & \equiv & \pi_1\pi_2(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : \mathbf{Nat} \\ u_1 & \equiv & \pi_1\pi_2\pi_2(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : d = m+n \\ h & \equiv & \pi_1\pi_2\pi_2\pi_2(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : \mathbf{M}_d \to \mathbf{M}_m \uplus \mathbf{M}_n \\ u_2 & \equiv & \pi_1\pi_2\pi_2\pi_2\pi_2(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : \mathbf{bijection}\,(h) \\ u_3 & \equiv & \pi_2\pi_2\pi_2\pi_2\pi_2(\mathcal{IH}(k)(\mathbf{p}_{19}(u'))(f)(q)) : (x:M(d)) \to \cdots \end{array} \tag{5.80}$$

We let C stand for the proposition in the consequence of (5.79).

$$\left[\begin{array}{l} m':\mathbf{Nat} \\ \left[\begin{array}{l} n':\mathbf{Nat} \\ \left[\begin{array}{l} \mathsf{suc}(d) = m'+n' \\ \left[\begin{array}{l} h': M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \left[\begin{array}{l} \mathbf{bijection}\,(h') \\ (x:M(\mathsf{suc}(d))) \to \left[\begin{array}{l} \left(\begin{array}{l} v_1: \left[\begin{array}{l} z:A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right] \end{array}\right) \to \left[\begin{array}{l} w:M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w) \end{array}\right] \\ \left(\begin{array}{l} v_2: \left[\begin{array}{l} z:B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right] \end{array}\right) \to \left[\begin{array}{l} w:M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right].$$

We will construct proof terms t_1, t_2, and t_3 that appear in the following delivation.

$$\dfrac{\begin{array}{ccccc} & & \overset{2}{\overline{\qquad\qquad}} & & \overset{2}{\overline{\qquad\qquad}} \\ \vdots\,\mathcal{D}_1 & & p' & & p' \\ t_1 : & & : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) \\ \quad =_{A\uplus B} \mathsf{inl}(z) \end{array}\right] & & : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) \\ \quad =_{A\uplus B} \mathsf{inr}(z) \end{array}\right] \\ \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) \\ \quad =_{A\uplus B} \mathsf{inl}(z) \end{array}\right] & & \vdots\,\mathcal{D}_2 & & \vdots\,\mathcal{D}_3 \\ \uplus \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) \\ \quad =_{A\uplus B} \mathsf{inr}(z) \end{array}\right] & \mathcal{T}_1 & t_2 : C & & t_3 : C \end{array}}{\mathsf{case}\ t_1\ \mathsf{of}\ (\lambda p'.t_2, \lambda p'.t_3) : C}\ \uplus E, 2$$

(5.81)

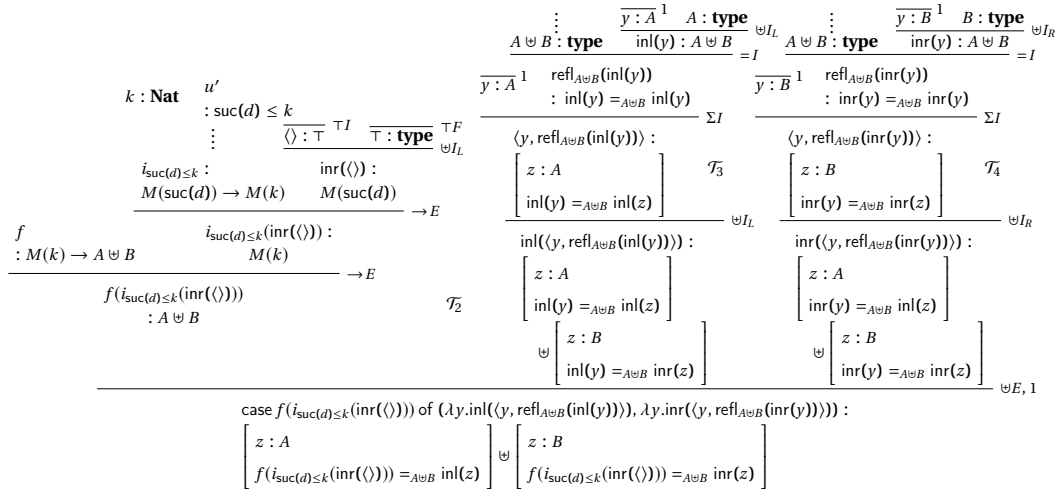
$\mathcal{T}_1$ is abbreviation for:

$$\lambda x.C : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right] \uplus \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle))) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right] \to \mathbf{type}$$

Henceforth, we will omit the derivation for the trivial case of type inference.

The intuition behind (5.81) is as follows: now we consider the case where the function's domain is extended from $\mathbf{M}_d$ to $\mathbf{M}_{\mathsf{suc}(d)}$. By the induction hypothesis, we already know which elements in $\mathbf{M}_d$ are related to A and which are to B. Thus, we only care about the newly-added element, $\mathsf{inr}(\langle\rangle)$. We again have two possibilities, whether it is related to A or B. This is shown in the derivation $\mathcal{D}_1$ below. We will then prove that we can obtain the consequence in either cases, in $\mathcal{D}_2$ and $\mathcal{D}_3$.

$\underline{\mathcal{D}_1}$:



Thus, t_1 in (5.81) is:

$$\mathsf{case}\ f(i_{\mathsf{suc}(d)\leq k}(\mathsf{inr}(\langle\rangle)))\ \mathsf{of}\ (\lambda y.\mathsf{inl}(\langle y, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(y))\rangle), \lambda y.\mathsf{inr}(\langle y, \mathsf{refl}_{A\uplus B}(\mathsf{inr}(y))\rangle)).$$

$\mathcal{T}_2$:

$$\lambda y.\begin{bmatrix} z : A \\ y =_{A\uplus B} \mathsf{inl}(z) \end{bmatrix} \uplus \begin{bmatrix} z : B \\ y =_{A\uplus B} \mathsf{inr}(z) \end{bmatrix} : A \uplus B \to \mathbf{type}$$

$\mathcal{T}_3$:

$$\begin{bmatrix} z : B \\ \mathsf{inl}(y) =_{A\uplus B} \mathsf{inr}(z) \end{bmatrix} : \mathbf{type}$$

$\mathcal{T}_4$:

$$\begin{bmatrix} z : A \\ \mathsf{inr}(y) =_{A\uplus B} \mathsf{inl}(z) \end{bmatrix} : \mathbf{type}$$

Next, we will give $\mathcal{D}_2$ in the derivation (5.81). For $\mathcal{D}_2$, we will show $\mathcal{D}_{2-1}$, $\mathcal{D}_{2-2}$, and $\mathcal{D}_{2-3}$ in

$$\dfrac{\dfrac{\begin{array}{l}\vdots\ \mathcal{D}_{2\text{–}2}\\ \langle t_{2\text{–}2\text{–}1}, t_{2\text{–}2\text{–}2}\rangle :\\ \textbf{bijection}\,(h')\end{array} \qquad \begin{array}{l}\vdots\ \mathcal{D}_{2\text{–}3}\\ t_{2\text{–}3} :\\ (x : M(\mathsf{suc}(d))) \to \left[\begin{array}{l}\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{array}\right]\\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{array}\right]\end{array}\right]\end{array}}{\begin{array}{l}\langle\langle t_{2\text{–}2\text{–}1}, t_{2\text{–}2\text{–}2}\rangle, t_{2\text{–}3}\rangle :\\ \left[\begin{array}{l}\textbf{bijection}\,(h')\\ (x : M(\mathsf{suc}(d))) \to \left[\begin{array}{l}\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{array}\right]\\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{array}\right]\end{array}\right]\end{array}\right]\end{array}}\times I}{h' : M(\mathsf{suc}(d)) \to \vdots\ M(\mathsf{suc}(m)) \uplus M(n)}\ \Sigma I$$

$$\left[\begin{array}{l}\langle h', \langle\langle t_{2\text{–}2\text{–}1}, t_{2\text{–}2\text{–}2}\rangle, t_{2\text{–}3}\rangle\rangle :\\ h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n')\\ \left[\begin{array}{l}\textbf{bijection}\,(h')\\ (x : M(\mathsf{suc}(d))) \to \left[\begin{array}{l}\left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{array}\right]\\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{array}\right]\right) \to \left[\begin{array}{l} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{array}\right]\end{array}\right]\end{array}\right]\end{array}\right] ,$$

(5.82)

with which we will be able to provide $\mathcal{D}_2$ by the following derivation.

$$
\dfrac{\dfrac{}{\mathbf{Nat}:\mathbf{type}} \quad \dfrac{}{d:\mathbf{Nat}} \quad \dfrac{\vdots}{m+n:\mathbf{Nat}} \quad \dfrac{\vdots}{\lambda x.\mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(x) : \mathbf{Nat}\to\mathbf{type}} \quad \dfrac{\vdots}{u_1 : d =_{\mathbf{Nat}} m+n} \quad \dfrac{\dfrac{}{\mathbf{Nat}:\mathbf{type}} \quad \dfrac{}{\mathsf{suc}(d):\mathbf{Nat}}}{\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d)) : \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(d)}{=_I}}{\mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))) : \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m+n)}\,(\text{Th15})
$$

$$
\dfrac{\mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))) : \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m+n) \qquad \dfrac{\vdots}{\mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m+n) \equiv_\beta \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m)+n}}{\mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))) : \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m)+n}\,conv
$$

$$
\dfrac{\mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))) : \mathsf{suc}(d) =_{\mathbf{Nat}} \mathsf{suc}(m)+n \qquad \vdots\,(5.82)\ \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle : \begin{bmatrix} h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \ldots \end{bmatrix}}{\langle \mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))), \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle\rangle : \begin{bmatrix} \mathsf{suc}(d) = m'+n' \\ \begin{bmatrix} h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \begin{bmatrix} \mathbf{bijection}(h') \\ (x : M(\mathsf{suc}(d))) \to \begin{bmatrix} \left(v_1 : \begin{bmatrix} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{bmatrix} \\ \left(v_2 : \begin{bmatrix} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}}\times I
$$

$$
\dfrac{\dfrac{\vdots}{n:\mathbf{Nat}} \qquad \langle \mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))), \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle\rangle : [\ldots]}{\langle n, \langle \mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))), \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle\rangle\rangle : \begin{bmatrix} n' : \mathbf{Nat} \\ \begin{bmatrix} \mathsf{suc}(d) = m'+n' \\ \begin{bmatrix} h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \begin{bmatrix} \mathbf{bijection}(h') \\ (x : M(\mathsf{suc}(d))) \to \begin{bmatrix} \left(v_1 : \begin{bmatrix} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{bmatrix} \\ \left(v_2 : \begin{bmatrix} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}}\Sigma I
$$

$$
\dfrac{\dfrac{\dfrac{\vdots}{m:\mathbf{Nat}}}{\mathsf{suc}(m):\mathbf{Nat}}\mathbf{Nat}I_s \qquad \langle n, \langle \mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))), \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle\rangle\rangle : [\ldots]}{\langle \mathsf{suc}(m), \langle n, \langle \mathbf{p}_{15}(\mathbf{p}_{13}(u_1);\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(d))), \langle h', \langle\langle t_{2-2-1}, t_{2-2-2}\rangle, t_{2-3}\rangle\rangle\rangle\rangle\rangle : \begin{bmatrix} m' : \mathbf{Nat} \\ \begin{bmatrix} n' : \mathbf{Nat} \\ \begin{bmatrix} \mathsf{suc}(d) = m'+n' \\ \begin{bmatrix} h' : M(\mathsf{suc}(d)) \to M(m') \uplus M(n') \\ \begin{bmatrix} \mathbf{bijection}(h') \\ (x : M(\mathsf{suc}(d))) \to \begin{bmatrix} \left(v_1 : \begin{bmatrix} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(m') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inl}(w)\end{bmatrix} \\ \left(v_2 : \begin{bmatrix} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : M(n') \\ h'(x) =_{M(m')\uplus M(n')} \mathsf{inr}(w)\end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}}\Sigma I
$$

(5.83)

$\mathcal{D}_{2-1}$:

$$
\dfrac{\dfrac{\vdots}{M(\mathsf{suc}(d)):\mathbf{type}} \qquad \dfrac{\dfrac{}{y : M(\mathsf{suc}(d))}{}^{2} \quad \mathcal{T}_5 \quad \dfrac{\dfrac{\vdots}{M(m):\mathbf{type}} \quad \dfrac{\vdots}{M(n):\mathbf{type}} \quad \dfrac{}{\top:\mathbf{type}}\top I \quad \dfrac{\dfrac{}{y' : M(d)}{}^{1} \quad \dfrac{\vdots}{h : M(d) \to M(m)\uplus M(n)}}{h(y') : M(m)\uplus M(n)}\to E}{\mathsf{case}\ h(y')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) : M(\mathsf{suc}(m))\uplus M(n)}(\text{Th12}) \quad \dfrac{\dfrac{\dfrac{}{y' : \top}{}^{1} \quad \dfrac{\vdots}{M(m):\mathbf{type}}}{\mathsf{inr}(y') : M(m)\uplus\top}\uplus I_R \quad \dfrac{\vdots}{M(n):\mathbf{type}}}{\mathsf{inl}(\mathsf{inr}(y')) : M(\mathsf{suc}(m))\uplus M(n)}\uplus I_L}{\mathsf{case}\ y\ \mathsf{of}\ (\lambda y'.\mathsf{case}\ h(y')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)), \lambda y'.\mathsf{inl}(\mathsf{inr}(y'))) : M(\mathsf{suc}(m))\uplus M(n)}\uplus E, 1}{\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda y'.\mathsf{case}\ h(y')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)), \lambda y'.\mathsf{inl}(\mathsf{inr}(y'))) : M(\mathsf{suc}(d)) \to M(\mathsf{suc}(m))\uplus M(n)}\Pi I, 2
$$

$\mathcal{T}_5$:

$$\lambda y.M(\mathsf{suc}(m)) \uplus M(n) : \ M(\mathsf{suc}(d)) \to \mathbf{type}$$

In what follows, h' stands for the obtained proof term

$$\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda y'.\mathsf{case}\ h(y')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)), \lambda y'.\mathsf{inl}(\mathsf{inr}(y'))).$$

$\underline{\mathcal{D}_{2-2}}$:

To give $\mathcal{D}_{2-2}$ in the derivation (5.82), we prove **bijection** (h'). We construct proof terms t_{2-2-1} and t_{2-2-2} in the following delivation.

$$\dfrac{\begin{array}{c}\vdots\ \mathcal{D}_{2-2-1}\\ t_{2-2-1} : \mathbf{injection}\,(h')\end{array} \quad \begin{array}{c}\vdots\ \mathcal{D}_{2-2-2}\\ t_{2-2-2} : \mathbf{surjection}\,(h')\end{array}}{\langle t_{2-2-1}, t_{2-2-2}\rangle : \mathbf{bijection}\,(h')}\times I$$

$\underline{\mathcal{D}_{2-2-1}}$:

For **injection** $(h') \equiv_\beta (y : M(\mathsf{suc}(d))) \to (y' : M(\mathsf{suc}(d))) \to h'(y) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to y =_{M(\mathsf{suc}(d))} y'$, we construct a proof term t_{2-2-1} by the following derivation.

$$\dfrac{\begin{array}{c}\dfrac{}{y : M(\mathsf{suc}(d))}{}^{1}\end{array} \quad \mathcal{T}_6 \quad \begin{array}{c}\vdots\\ M(\mathsf{suc}(d))\\ : \mathbf{type}\end{array} \quad \dfrac{\dfrac{\begin{array}{c}\dfrac{}{y' : M(\mathsf{suc}(d))}{}^{3}\end{array} \quad \mathcal{T}_7 \quad \begin{array}{c}\overline{y : M(d)}^{\,2}\ \ \overline{y' : M(d)}^{\,4}\\ \vdots\ \mathcal{D}_{2-2-1(1)}\\ t_{2-2-1(1)} :\\ h'(\mathsf{inl}(y)) = h'(\mathsf{inl}(y'))\\ \to \mathsf{inl}(y) = \mathsf{inl}(y')\end{array} \quad \begin{array}{c}\overline{y : M(d)}^{\,2}\ \ \overline{y' : \top}^{\,4}\\ \vdots\ \mathcal{D}_{2-2-1(2)}\\ t_{2-2-1(2)} :\\ h'(\mathsf{inl}(y)) = h'(\mathsf{inr}(y'))\\ \to \mathsf{inl}(y) = \mathsf{inr}(y')\end{array}}{\begin{array}{c}\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y'.t_{2-2-1(1)}, \lambda y'.t_{2-2-1(2)}) :\\ h'(\mathsf{inl}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y')\\ \to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} y'\end{array}}\uplus E, 4 \quad \begin{array}{c}\vdots\\ M(\mathsf{suc}(d))\\ : \mathbf{type}\end{array}}{\begin{array}{c}\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(1)}, \lambda y''.t_{2-2-1(2)}) :\\ (y' : M(\mathsf{suc}(d)))\\ \to h'(\mathsf{inl}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} y'\end{array}}\Pi I, 3 \quad \dfrac{\dfrac{\begin{array}{c}\dfrac{}{y' : M(\mathsf{suc}(d))}{}^{5}\end{array} \quad \mathcal{T}_8 \quad \begin{array}{c}\overline{y : \top}^{\,2}\ \ \overline{y' : M(d)}^{\,6}\\ \vdots\ \mathcal{D}_{2-2-1(3)}\\ t_{2-2-1(3)} :\\ h'(\mathsf{inr}(y)) = h'(\mathsf{inl}(y'))\\ \to \mathsf{inr}(y) = \mathsf{inl}(y')\end{array} \quad \begin{array}{c}\overline{y : \top}^{\,2}\ \ \overline{y' : \top}^{\,6}\\ \vdots\ \mathcal{D}_{2-2-1(4)}\\ t_{2-2-1(4)} :\\ h'(\mathsf{inr}(y)) = h'(\mathsf{inr}(y'))\\ \to \mathsf{inr}(y) = \mathsf{inr}(y')\end{array}}{\begin{array}{c}\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y'.t_{2-2-1(3)}, \lambda y'.t_{2-2-1(4)}) :\\ h'(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y')\\ \to \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} y'\end{array}}\uplus E, 6}{\begin{array}{c}\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(3)}, \lambda y''.t_{2-2-1(4)}) :\\ (y' : M(\mathsf{suc}(d)))\\ \to h'(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} y'\end{array}}\Pi I, 5}{\begin{array}{c}\mathsf{case}\ y\ \mathsf{of}\ (\lambda y.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(1)}, \lambda y''.t_{2-2-1(2)}), \lambda y.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(3)}, \lambda y''.t_{2-2-1(4)})) :\\ (y' : M(\mathsf{suc}(d)))\\ \to h'(y) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to y =_{M(\mathsf{suc}(d))} y'\end{array}}\uplus E, 2$$

$$\dfrac{\begin{array}{c}\vdots\\ M(\mathsf{suc}(d))\\ : \mathbf{type}\end{array} \quad \cdots}{\begin{array}{c}\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda y'''.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(1)}, \lambda y''.t_{2-2-1(2)}), \lambda y'''.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2-2-1(3)}, \lambda y''.t_{2-2-1(4)})) :\\ (y : M(\mathsf{suc}(d))) \to (y' : M(\mathsf{suc}(d))) \to h'(y) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to y =_{M(\mathsf{suc}(d))} y'\end{array}}\Pi I, 1 ,$$

(5.84)

$\mathcal{T}_6$:

$$\lambda y.(y' : M(\mathsf{suc}(d))) \to h'(y) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to y =_{M(\mathsf{suc}(d))} y' : M(\mathsf{suc}(d)) \to \mathbf{type}$$

$\mathcal{T}_7$:

$$\lambda y'.h'(\mathsf{inl}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} y' : M(\mathsf{suc}(d)) \to \mathbf{type}$$

$\mathcal{T}_8$:

$$\lambda y'.h'(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y') \to \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} y' : M(\mathsf{suc}(d)) \to \mathbf{type}$$

The following term that is at the bottom of (5.84) corresponds to $t_{2\text{–}2\text{–}1}$.

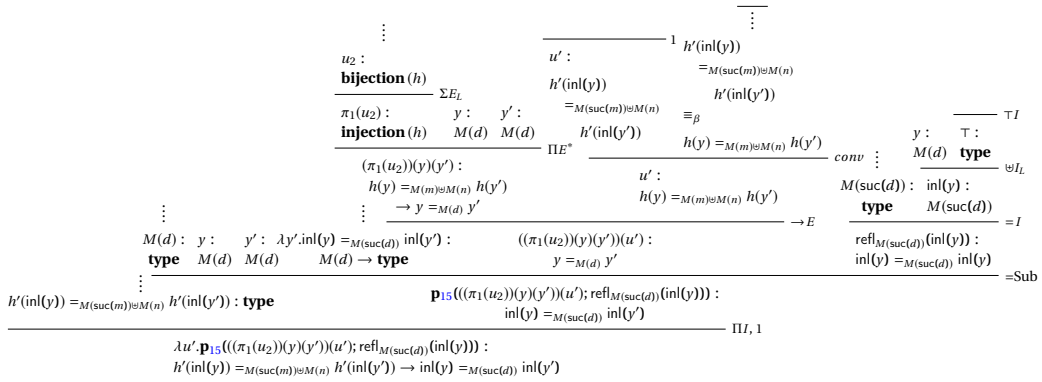
$$\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda y'''.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2\text{–}2\text{–}1(1)}, \lambda y''.t_{2\text{–}2\text{–}1(2)}), \lambda y'''.\lambda y'.\mathsf{case}\ y'\ \mathsf{of}\ (\lambda y''.t_{2\text{–}2\text{–}1(3)}, \lambda y''.t_{2\text{–}2\text{–}1(4)}))) \tag{5.85}$$

As the assumptions $y : \mathbf{M}_{\mathsf{suc}(d)}$ and $y' : \mathbf{M}_{\mathsf{suc}(d)}$ are both disjoint union of types, the proof is devided into the following four cases with respect to the assumptions.

- $y : \mathbf{M}_d$ and $y' : \mathbf{M}_d$ (derivation $\mathcal{D}_{2\text{–}2\text{–}1(1)}$ for $t_{2\text{–}2\text{–}1(1)}$)
- $y : \mathbf{M}_d$ and $y' : \top$ (derivation $\mathcal{D}_{2\text{–}2\text{–}1(2)}$ for $t_{2\text{–}2\text{–}1(2)}$)
- $y : \top$ and $y' : \mathbf{M}_d$ (derivation $\mathcal{D}_{2\text{–}2\text{–}1(3)}$ for $t_{2\text{–}2\text{–}1(3)}$)
- $y : \top$ and $y' : \top$ (derivation $\mathcal{D}_{2\text{–}2\text{–}1(4)}$ for $t_{2\text{–}2\text{–}1(4)}$)

Below, we prove the four cases one-by-one.

$\underline{\mathcal{D}_{2\text{–}2\text{–}1(1)}}$:



$\mathcal{D}_{2-2-1(2)}$:

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{\vdots}{h : M(d) \to M(m) \uplus M(n)} \quad y : M(d)
}{h(y) : M(m) \uplus M(n)}\,\Pi E
\quad \mathcal{T}_9
\quad
\begin{array}{c}
\overline{x : M(m)}^{\,1} \quad y' : \top \\
\vdots\ \mathcal{D}_a \\
\mathbf{a} : \\
\mathsf{case}\ \mathsf{inl}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\
=_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
\quad
\begin{array}{c}
\overline{x : M(n)}^{\,1} \quad y' : \top \\
\vdots\ \mathcal{D}_b \\
\mathbf{b} : \\
\mathsf{case}\ \mathsf{inr}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\
=_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}{
\begin{array}{c}
\mathsf{case}\ h(y)\ \mathsf{of}\ (\lambda x.\mathbf{a}, \lambda x.\mathbf{b}) : \\
\mathsf{case}\ h(y)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,\uplus E, 1
\quad
\begin{array}{c}
\overline{} \\
\vdots \\
\mathsf{case}\ h(y)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\
=_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y') \\
\equiv_\beta h'(\mathsf{inl}(y)) =_{M(\mathsf{suc}(m)) \uplus M(n)} h'(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}{
\begin{array}{c}
\mathsf{case}\ h(y)\ \mathsf{of}\ (\lambda x.\mathbf{a}, \lambda x.\mathbf{b}) : \\
h'(\mathsf{inl}(y)) =_{M(\mathsf{suc}(m)) \uplus M(n)} h'(\mathsf{inr}(y')) \to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,conv
\quad , \tag{5.86}
$$

where $\mathcal{D}_a$ and $\mathcal{D}_b$ are given below.

$\mathcal{T}_9$:

$$\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y') : M(m) \uplus M(n) \to \mathbf{type}$$

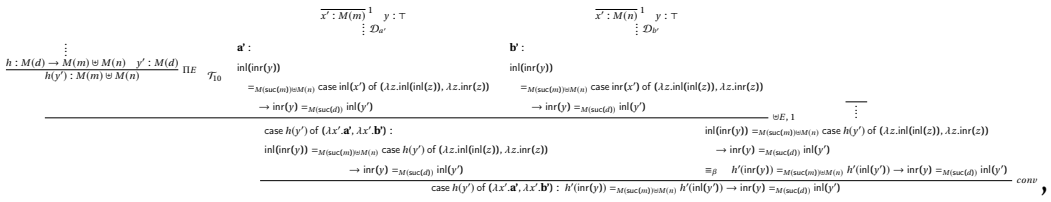
$\mathcal{D}_a$ in $\mathcal{D}_{2-2-1(2)}$:

$$
\dfrac{
\begin{array}{c}\vdots \\ \mathsf{case}\ \mathsf{inl}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) : \mathbf{type}\end{array}
\quad
\dfrac{
\dfrac{
\dfrac{
\dfrac{}{\mathbf{p}_{20} : \mathbf{injection}\,(\lambda x.\mathsf{inl}(x))}\,(\text{Lm20})
\quad
\begin{array}{c}\vdots \\ M(\mathsf{suc}(m)) : \mathbf{type}\end{array}
\quad
\begin{array}{c}\vdots \\ M(n) : \mathbf{type}\end{array}
\quad
\dfrac{x : M(m) \quad \overline{\top : \mathbf{type}}}{\mathsf{inl}(x) : M(\mathsf{suc}(m))}\,\uplus I_L
\quad
\dfrac{\begin{array}{c}\vdots \\ M(m) : \mathbf{type}\end{array} \quad y' : \top}{\mathsf{inr}(y') : M(\mathsf{suc}(m))}\,\uplus I_R
}{
\begin{array}{c}
\mathbf{p}_{20}(M(\mathsf{suc}(m))(M(n))(\mathsf{inl}(x))(\mathsf{inr}(y')) : \\
\mathsf{inl}(\mathsf{inl}(x)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(x) =_{M(\mathsf{suc}(m))} \mathsf{inr}(y')
\end{array}
}\,\Pi E^*
\quad
\dfrac{
\dfrac{}{\begin{array}{c}u' : \\ \mathsf{case}\ \mathsf{inl}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))\end{array}}\,2
\quad
\begin{array}{c}
\overline{} \\ \vdots \\
\mathsf{case}\ \mathsf{inl}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\
=_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\equiv_\beta \\
\mathsf{inl}(\mathsf{inl}(x)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))
\end{array}
}{u' : \mathsf{inl}(\mathsf{inl}(x)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))}\,conv
}{
\begin{array}{c}
(\mathbf{p}_{20}(M(\mathsf{suc}(m))(M(n))(\mathsf{inl}(x))(\mathsf{inr}(y')))(u') : \\
\mathsf{inl}(x) =_{M(\mathsf{suc}(m))} \mathsf{inr}(y')
\end{array}
}\,{\to}E
\quad
\dfrac{
\begin{array}{c}\vdots \\ M(m) : \mathbf{type}\end{array} \quad \overline{\top : \mathbf{type}}^{\,\top I} \quad x : M(m) \quad y' : \top
}{
\begin{array}{c}
\mathbf{p}_{25}(x; y') : \\
\neg(\mathsf{inl}(x) =_{M(\mathsf{suc}(m))} \mathsf{inr}(y'))
\end{array}
}\,(\text{Th25})
}{
\begin{array}{c}
(\mathbf{p}_{25}(x; y'))\big((\mathbf{p}_{20}(M(\mathsf{suc}(m))(M(n))(\mathsf{inl}(x))(\mathsf{inr}(y')))(u')\big) : \\
\bot
\end{array}
}\,{\to}E
\quad
\begin{array}{c}\vdots \\ \lambda b.\mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y') : \\ \bot \to \mathbf{type}\end{array}
}{
\begin{array}{c}
\mathsf{case}_{(\mathbf{p}_{25}(x;y'))\left((\mathbf{p}_{20}(M(\mathsf{suc}(m)))(M(n))(\mathsf{inl}(x))(\mathsf{inr}(y')))(u')\right)}() : \\
\mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,\bot E
}{
\begin{array}{c}
\lambda u'.\mathsf{case}_{(\mathbf{p}_{25}(x;y'))\left((\mathbf{p}_{20}(M(\mathsf{suc}(m)))(M(n))(\mathsf{inl}(x))(\mathsf{inr}(y')))(u')\right)}() : \\
\mathsf{case}\ \mathsf{inl}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,\Pi I, 2
$$

$\mathcal{D}_b$ in $\mathcal{D}_{2-2-1(2)}$:

$$
\dfrac{
\begin{array}{c}\vdots \\ \mathsf{case}\ \mathsf{inr}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) : \mathbf{type}\end{array}
\quad
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{}{\begin{array}{c}u' : \\ \mathsf{case}\ \mathsf{inr}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))\end{array}}\,3
\quad
\begin{array}{c}
\overline{} \\ \vdots \\
\mathsf{case}\ \mathsf{inr}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\
=_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\equiv_\beta \\
\mathsf{inr}(x) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))
\end{array}
}{\begin{array}{c}u' : \\ \mathsf{inr}(x) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y'))\end{array}}\,conv
}{
\begin{array}{c}
\mathbf{p}_{13}(u') : \\
\mathsf{inl}(\mathsf{inr}(y')) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inr}(x)
\end{array}
}\,(\text{Th13})
\quad
\dfrac{
\begin{array}{c}\vdots \\ M(\mathsf{suc}(m)) : \mathbf{type}\end{array}
\quad
\begin{array}{c}\vdots \\ M(n) : \mathbf{type}\end{array}
\quad
\dfrac{\begin{array}{c}\vdots \\ M(m) : \mathbf{type}\end{array} \quad y' : \top}{\mathsf{inr}(y') : M(\mathsf{suc}(m))}\,\uplus I_R
\quad x : M(n)
}{
\begin{array}{c}
\mathbf{p}_{25}(\mathsf{inr}(y'); x) : \\
\neg(\mathsf{inl}(\mathsf{inr}(y')) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inr}(x))
\end{array}
}\,(\text{Th25})
}{
\begin{array}{c}
(\mathbf{p}_{25}(\mathsf{inr}(y'); x))(\mathbf{p}_{13}(u')) : \\
\bot
\end{array}
}\,{\to}E
\quad
\begin{array}{c}\vdots \\ \lambda b.\mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y') : \\ \bot \to \mathbf{type}\end{array}
}{
\begin{array}{c}
\mathsf{case}_{(\mathbf{p}_{25}(\mathsf{inr}(y');x))(\mathbf{p}_{13}(u'))}() : \\
\mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,\bot E
}{
\begin{array}{c}
\lambda u'.\mathsf{case}_{(\mathbf{p}_{25}(\mathsf{inr}(y');x))(\mathbf{p}_{13}(u'))}() : \\
\mathsf{case}\ \mathsf{inr}(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m)) \uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \\
\to \mathsf{inl}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y')
\end{array}
}\,\Pi I, 3
$$

$\underline{\mathcal{D}_{2\text{–}2\text{–}1(3)}}$:



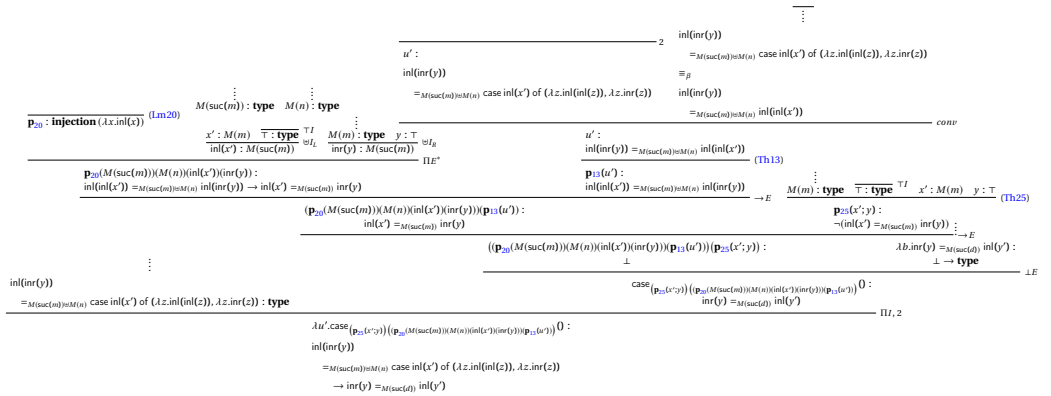
(5.87)

where $\mathcal{D}_{a'}$ and $\mathcal{D}_{b'}$ are given below.

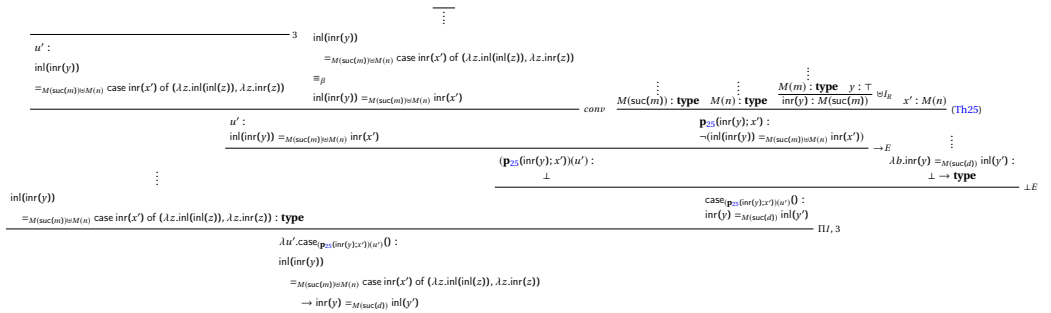
$\underline{\mathcal{T}_{10}}$:

$\lambda y.\mathsf{inl}(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{case}\ y\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \to \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} \mathsf{inl}(y') : M(m) \uplus M(n) \to \mathbf{type}$

$\underline{\mathcal{D}_{a'}}$ in $\underline{\mathcal{D}_{2\text{–}2\text{–}1(3)}}$:



$\underline{\mathcal{D}_{b'}}$ in $\underline{\mathcal{D}_{2\text{–}2\text{–}1(2)}}$:



$\underline{\mathcal{D}_{2\text{–}2\text{–}1(4)}}$:

Given $y : \top$ and $y' : \top$, assume $u' : h'(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inr}(y'))$. Since $h'(\mathsf{inr}(y)) \equiv_\beta \mathsf{inl}(\mathsf{inr}(y))$ and $h'(\mathsf{inr}(y')) \equiv_\beta \mathsf{inl}(\mathsf{inr}(y'))$, we have the fol-

lowing by $conv$.

$$u' : \mathsf{inl}(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(\mathsf{inr}(y')) \tag{5.88}$$

By Lemma 23, we have proof $\mathbf{p}_{23} : \mathbf{injection}\,(\lambda y.\mathsf{inl}(\mathsf{inr}(y)))$. Therefore, we obtain

$$\mathbf{p}_{23}(M(m))(\top)(M(n))(y)(y')(u') : y =_{\top} y', \tag{5.89}$$

by applying ΠE several times. Now, by applying $=I$ with $\mathsf{inr}(y) : M(\mathsf{suc}(d))$, we have

$$\mathsf{refl}_{M(\mathsf{suc}(d))}(\mathsf{inr}(y)) : \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y). \tag{5.90}$$

Thus, by substitution by identity (5.89) and (5.90), we obtain

$$\begin{gathered}\mathbf{p}_{15}(\mathbf{p}_{23}(M(m))(\top)(M(n))(y)(y')(u'); \mathsf{refl}_{M(\mathsf{suc}(d))}(\mathsf{inr}(y))) : \\ \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y'),\end{gathered} \tag{5.91}$$

from which we obtain the following by ΠI with u'.

$$\begin{gathered}\lambda u'.\mathbf{p}_{15}(\mathbf{p}_{23}(M(m))(\top)(M(n))(y)(y')(u'); \mathsf{refl}_{M(\mathsf{suc}(d))}(\mathsf{inr}(y))) : \\ h'(\mathsf{inr}(y)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inr}(y')) \to \mathsf{inr}(y) =_{M(\mathsf{suc}(d))} \mathsf{inr}(y').\end{gathered} \tag{5.92}$$

$\underline{\mathcal{D}_{2\text{–}2\text{–}2}}$ in $\mathcal{D}_2$ at (5.82):

Next, we prove **surjection** (h'). As **surjection** $(h') \equiv (w : M(\mathsf{suc}(m)) \uplus M(n)) \to \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ w =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]$, the proof $t_{2\text{–}2\text{–}2}$ in (5.82) is constructed by the following derivation.

$$\dfrac{\dfrac{M(\mathsf{suc}(m)) \uplus M(n) : \mathbf{type} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{w : M(\mathsf{suc}(m)) \uplus M(n)}\,1 \quad \mathcal{T}_{11} \qquad \dfrac{\dfrac{}{w' : M(\mathsf{suc}(m))}\,2 \quad \mathcal{T}_{12} \qquad \dfrac{\dfrac{}{w'' : M(m)}\,3}{\vdots\ \mathcal{D}_{2\text{–}2\text{–}2(1)}}}{\left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inl}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]^{t_{2\text{–}2\text{–}2(1)}}} \qquad \dfrac{\dfrac{}{w'' : \top}\,3}{\vdots\ \mathcal{D}_{2\text{–}2\text{–}2(2)}}}{\left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inr}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]^{t_{2\text{–}2\text{–}2(2)}}}}{\mathsf{case}\ w'\ \mathsf{of}\ (\lambda w''.t_{2\text{–}2\text{–}2(1)}, \lambda w''.t_{2\text{–}2\text{–}2(2)}) : \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(w') =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]}\uplus E, 3 \qquad \dfrac{\dfrac{}{w' : M(n)}\,2}{\vdots\ \mathcal{D}_{2\text{–}2\text{–}2(3)}}}{\mathsf{case}\ w\ \mathsf{of}\ (\lambda w'.\mathsf{case}\ w'\ \mathsf{of}\ (\lambda w''.t_{2\text{–}2\text{–}2(1)}, \lambda w''.t_{2\text{–}2\text{–}2(2)}), \lambda w'.t_{2\text{–}2\text{–}2(3)}) : \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ w =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]}\uplus E, 2}{\lambda w.\mathsf{case}\ w\ \mathsf{of}\ (\lambda w'.\mathsf{case}\ w'\ \mathsf{of}\ (\lambda w''.t_{2\text{–}2\text{–}2(1)}, \lambda w''.t_{2\text{–}2\text{–}2(2)}), \lambda w'.t_{2\text{–}2\text{–}2(3)}) : (w : M(\mathsf{suc}(m)) \uplus M(n)) \to \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ w =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]}\Pi I, 1$$

As $\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n \equiv (\mathbf{M}_m \uplus \top) \uplus \mathbf{M}_n$, the proof is devided into three cases. When we assume $\mathbf{M}_m$ or $\mathbf{M}_n$, the proof is given by IH. In case of $\langle\rangle : \top$, as this is the one to be added to the $\mathbf{M}_m$ side, it should be associated with the $\mathsf{suc}(d)$-th element of $\mathbf{M}_{\mathsf{suc}(d)}$.

$\mathcal{D}_{2\text{–}2\text{–}2(1)}$, $\mathcal{D}_{2\text{–}2\text{–}2(2)}$, and $\mathcal{D}_{2\text{–}2\text{–}2(3)}$ are given below.

$\mathcal{T}_{11}$:

$$\lambda w. \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ w =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right] : M(\mathsf{suc}(m)) \uplus M(n) \to \mathbf{type}$$

$\mathcal{T}_{12}$:

$$\lambda w'. \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(w') =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right] : M(\mathsf{suc}(m)) \to \mathbf{type}$$

$\underline{\mathcal{D}_{2\text{–}2\text{–}2(1)}}$ in $\mathcal{D}_{2\text{–}2\text{–}2}$:

Given $w'' : M(m)$, we prove $\left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inl}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right]$. From the assumption, we get

$$\mathsf{inl}(w'') : M(m) \uplus M(n), \tag{5.93}$$

by $\uplus I$. From $\mathcal{IH}$ and assumptions $k : \mathbf{Nat}$, $\mathbf{p}_{19}(u) : d \le k$ (by Theorem 19 and the assumption $u : \mathsf{suc}(d) \le k$), $f : M(k) \to A \uplus B$, and $q :$ **bijection** (f), by applying ΠE several times and then applying ΣE several times, we obtain

$$\pi_2(u_2) : \mathbf{surjection}\,(h)\,. \tag{5.94}$$

By applying ΠE with (5.94) and (5.93), we have

$$(\pi_2(u_2))(\mathsf{inl}(w'')) : \left[\begin{array}{l} y' : M(d) \\ \mathsf{inl}(w'') =_{M(m)\uplus M(n)} h(y') \end{array}\right]. \tag{5.95}$$

Let $\mathbf{s}$ be $(\pi_2(u_2))(\mathsf{inl}(w''))$. By applying ΣE_L and ΣE_R with (5.95), respectively, we obtain:

$$\pi_1(\mathbf{s}) \quad : \quad M(d) \tag{5.96}$$

$$\pi_2(\mathbf{s}) \quad : \quad \mathsf{inl}(w'') =_{M(m)\uplus M(n)} h(\pi_1(\mathbf{s})). \tag{5.97}$$

We have $\mathsf{inl}(\pi_1(\mathbf{s})) : M(\mathsf{suc}(d))$ by applying $\uplus I$ with (5.96). By ΠE with h' and $\mathsf{inl}(\pi_1(\mathbf{s}))$, we have

$$h'(\mathsf{inl}(\pi_1(\mathbf{s}))) : M(\mathsf{suc}(m)) \uplus M(n), \tag{5.98}$$

from which, by $=I$, we obtain

$$\begin{aligned} &\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s})))) : \\ &h'(\mathsf{inl}(\pi_1(\mathbf{s}))) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s}))). \end{aligned} \tag{5.99}$$

As $h'(\mathsf{inl}(\pi_1(\mathbf{s}))) \equiv_\beta \mathsf{case}\ h(\pi_1(\mathbf{s}))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z))$, we have

$$\begin{aligned} &\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s})))) : \\ &\mathsf{case}\ h(\pi_1(\mathbf{s}))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s}))). \end{aligned} \tag{5.100}$$

From (5.97) and symmetric property of identity (Theorem 13), we have $\mathbf{p}_{13}(\pi_2(\mathbf{s})) : h(\pi_1(\mathbf{s})) =_{M(m)\uplus M(n)} \mathsf{inl}(w'')$. By (5.100) and substitution by identity (Theorem 15), $\mathbf{p}_{13}(\pi_2(\mathbf{s}))$, we obtain

$$\begin{aligned} &\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s})))) : \\ &\mathsf{case}\ \mathsf{inl}(w'')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s}))). \end{aligned} \tag{5.101}$$

As $\mathsf{case}\ \mathsf{inl}(w'')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \equiv_\beta \mathsf{inl}(\mathsf{inl}(w''))$ holds, we have

$$\begin{aligned} &\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s})))) : \\ &\mathsf{inl}(\mathsf{inl}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s}))). \end{aligned} \tag{5.102}$$

Thus, by ΣI with $\mathsf{inl}(\pi_1(\mathbf{s})) : M(\mathsf{suc}(d))$ and (5.102), we obtain

$$\langle \mathsf{inl}(\pi_1(\mathbf{s})), \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s})))) \rangle : \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inl}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array} \right] \tag{5.103}$$

$\underline{\mathcal{D}_{2\text{–}2\text{–}2(2)}}$ in $\mathcal{D}_{2\text{–}2\text{–}2}$:

Suppose we have $w'' : \top$. We prove $\left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inr}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array} \right]$.

From h' and $\mathsf{inr}(\langle\rangle) : M(\mathsf{suc}(d))$, by $=I$, we have

$$\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inr}(\langle\rangle))) : h'(\mathsf{inr}(\langle\rangle)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inr}(\langle\rangle)). \tag{5.104}$$

Since $h'(\mathsf{inr}(\langle\rangle)) \equiv_\beta \mathsf{inl}(\mathsf{inr}(\langle\rangle))$ holds, by β-conversion and $conv$, we obtain

$$\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inr}(\langle\rangle))) : \mathsf{inl}(\mathsf{inr}(\langle\rangle)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inr}(\langle\rangle)). \tag{5.105}$$

By applying $\top E$ with the given $w'' : \top$ and (5.105), we obtain

$$\mathsf{case}_{w''}(\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inr}(\langle\rangle)))) : \mathsf{inl}(\mathsf{inr}(w')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inr}(\langle\rangle)) \tag{5.106}$$

From $\mathsf{inr}(\langle\rangle) : M(\mathsf{suc}(d))$ and (5.106), by ΣI, we obtain

$$\langle \mathsf{inr}(\langle\rangle), \mathsf{case}_{w''}(\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inr}(\langle\rangle)))) \rangle : \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inl}(\mathsf{inr}(w'')) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array} \right] \tag{5.107}$$

$\underline{\mathcal{D}_{2\text{–}2\text{–}2(3)}}$:

The proof is similar to $\mathcal{D}_{2\text{–}2\text{–}2(1)}$.

Given $w'' : M(n)$, we prove $\left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inr}(w') =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array} \right]$. From the assumption, we get

$$\mathsf{inr}(w'') : M(m) \uplus M(n), \tag{5.108}$$

by $\uplus I$. From $\mathcal{IH}$ and assumptions $k : \mathbf{Nat}$, $\mathbf{p}_{19}(u) : d \leq k$ (by Theorem 19 and the assumption $u : \mathsf{suc}(d) \leq k$), $f : M(k) \to A \uplus B$, and $q :$ **bijection** (f),

by applying ΠE several times and then applying ΣE several times, we obtain

$$\pi_2(u_2) : \textbf{surjection}\,(h)\,. \tag{5.109}$$

By applying ΠE with (5.109) and (5.108), we have

$$(\pi_2(u_2))(\mathsf{inr}(w'')) : \left[\begin{array}{l} y' : M(d) \\ \mathsf{inr}(w'') =_{M(m)\uplus M(n)} h(y') \end{array}\right]. \tag{5.110}$$

Let $\mathbf{s'}$ be $(\pi_2(u_2))(\mathsf{inr}(w''))$. By applying ΣE_L and ΣE_R with (5.110), respectively, we obtain:

$$\pi_1(\mathbf{s'}) \;:\; M(d) \tag{5.111}$$

$$\pi_2(\mathbf{s'}) \;:\; \mathsf{inr}(w'') =_{M(m)\uplus M(n)} h(\pi_1(\mathbf{s'})). \tag{5.112}$$

We have $\mathsf{inl}(\pi_1(\mathbf{s'})) : M(\mathsf{suc}(d))$ by applying $\uplus I$ with (5.111). By ΠE with h' and $\mathsf{inl}(\pi_1(\mathbf{s'}))$, we have

$$h'(\mathsf{inl}(\pi_1(\mathbf{s'}))) : M(\mathsf{suc}(m)) \uplus M(n), \tag{5.113}$$

from which, by $=I$, we obtain

$$\begin{array}{l} \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s'})))) : \\ h'(\mathsf{inl}(\pi_1(\mathbf{s'}))) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s'}))). \end{array} \tag{5.114}$$

As $h'(\mathsf{inl}(\pi_1(\mathbf{s'}))) \equiv_\beta \mathsf{case}\; h(\pi_1(\mathbf{s'}))\; \mathsf{of}\; (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z))$, we have

$$\begin{array}{l} \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s'})))) : \\ \mathsf{case}\; h(\pi_1(\mathbf{s'}))\; \mathsf{of}\; (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s'}))). \end{array} \tag{5.115}$$

From (5.112) and symmetric propery of identity (Theorem 13), we have $\mathbf{p}_{13}(\pi_2(\mathbf{s'})) \;:\; h(\pi_1(\mathbf{s'})) =_{M(m)\uplus M(n)} \mathsf{inr}(w'')$. By (5.115) and substitution by identity (Theorem 15), $\mathbf{p}_{13}(\pi_2(\mathbf{s'}))$, we obtain

$$\begin{array}{l} \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s'})))) : \\ \mathsf{case}\; \mathsf{inr}(w'')\; \mathsf{of}\; (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s'}))). \end{array} \tag{5.116}$$

As case $\mathsf{inr}(w'')$ of $(\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \equiv_\beta \mathsf{inr}(w'')$ holds, we have

$$\begin{aligned}&\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s'}))))\ :\\ &\mathsf{inr}(w'') =_{M(\mathsf{suc}(m))\uplus M(n)} h'(\mathsf{inl}(\pi_1(\mathbf{s'}))).\end{aligned} \tag{5.117}$$

Thus, by ΣI with $\mathsf{inl}(\pi_1(\mathbf{s'})) : M(\mathsf{suc}(d))$ and (5.117), we obtain

$$\langle \mathsf{inl}(\pi_1(\mathbf{s'})), \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(h'(\mathsf{inl}(\pi_1(\mathbf{s'}))))\rangle : \left[\begin{array}{l} y : M(\mathsf{suc}(d)) \\ \mathsf{inr}(w'') =_{M(\mathsf{suc}(m))\uplus M(n)} h'(y) \end{array}\right] \tag{5.118}$$

Finally, we prove $\underline{\mathcal{D}_{2\text{–}3}}$ in $\mathcal{D}_2$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\overline{x' : M(d)}^{\,1}}\ \mathcal{D}_{2\text{–}3\text{–}1} \qquad \dfrac{}{\overline{x' : \top}^{\,1}}\ p' : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(\mathsf{inr}(\langle\rangle))) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\ \mathcal{D}_{2\text{–}3\text{–}2}}{\overline{x : M(\mathsf{suc}(d))}^{\,2}\quad \mathcal{T}_{13} \quad \left[\begin{array}{l} t_{2\text{–}3\text{–}1} : \left[\begin{array}{l} \left(v_1' : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(\mathsf{inl}(x'))) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(\mathsf{suc}(m)) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2' : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(\mathsf{inl}(x'))) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \quad \left[\begin{array}{l} t_{2\text{–}3\text{–}2} : \left[\begin{array}{l} \left(v_1' : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(\mathsf{inr}(x'))) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(\mathsf{suc}(m)) \\ h'(\mathsf{inr}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2' : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(\mathsf{inr}(x'))) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h'(\mathsf{inr}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right]}{\vdots\ M(\mathsf{suc}(d)) : \mathbf{type} \qquad \mathsf{case}\ x\ \mathsf{of}\ (\lambda x'.t_{2\text{–}3\text{–}1}, \lambda x'.t_{2\text{–}3\text{–}2}) : \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(\mathsf{suc}(m)) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right]}\ \Pi I, 2 \\ \lambda x.\mathsf{case}\ x\ \mathsf{of}\ (\lambda x'.t_{2\text{–}3\text{–}1}, \lambda x'.t_{2\text{–}3\text{–}2}) : \\ (x : M(\mathsf{suc}(d))) \to \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(\mathsf{suc}(m)) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right]$$

The proof $t_{2\text{–}3}$ is given at the bottom of the derivation.

The left hand side can be proved by IH. For the right hand side, it leads to contradiction, because at the beginning of the derivation $\mathcal{D}_2$, we have assumed that the newly added element is associated with A, not B.

$\mathcal{D}_{2\text{–}3\text{–}1}$ and $\mathcal{D}_{2\text{–}3\text{–}2}$ are given below.

$\mathcal{T}_{13}$:

$$\lambda x.\left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(\mathsf{suc}(m)) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h'(x) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] : M(\mathsf{suc}(d)) \to \mathbf{type}$$

$\underline{\mathcal{D}_{2\text{–}3\text{–}1}}$:

Given $x' : M(d)$, by applying ΠE with u_3 of (5.80), we obtain

$$u_3(x') : \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f\big((\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x')\big) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f\big((\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x')\big) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \tag{5.119}$$

By Lemma 27, we obtain an identity by the following delivation, $\mathcal{D}_{2\text{–}3\text{–}1(1)}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{p}_{27} : (k,l,n : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l)+n) \to (y : M(l)) \to (\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle n, p_1\rangle)))(\mathsf{inl}(y)) =_{M(k)} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle n, p_1\rangle))))(y)}\,(\text{Lm27}) \quad k : \mathbf{Nat} \quad d : \mathbf{Nat} \quad \dfrac{u' : \mathsf{suc}(d) \le k}{\pi_1(u') : \mathbf{Nat}}\,\Sigma E_L \quad \dfrac{u' : \mathsf{suc}(d) \le k}{\pi_2(u') : k =_{\mathbf{Nat}} \mathsf{suc}(d) + \pi_1(u')}\,\Sigma E_R \quad x' : M(d)}{\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x') : (\pi_1(\mathbf{i}(k)(\mathsf{suc}(d))(\langle \pi_1 u', \pi_2 u'\rangle)))(\mathsf{inl}(x')) =_{M(k)} (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(\langle \pi_1 u', \pi_2 u'\rangle))))(x')}\,\Pi E^* \quad \begin{array}{l} u' \equiv \langle \pi_1 u', \pi_2 u'\rangle \\ \vdots \\ (\pi_1(\mathbf{i}(k)(\mathsf{suc}(d))(\langle \pi_1 u', \pi_2 u'\rangle)))(\mathsf{inl}(x')) \\ =_{M(k)} (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(\langle \pi_1 u', \pi_2 u'\rangle))))(x') \\ \equiv_{\beta} \\ (i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x')) \\ =_{M(k)} (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x') \end{array}\,conv}{\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x') : (i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x')) =_{M(k)} (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x')}\,(\text{Th13})}{\mathbf{p}_{13}(\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x')) : (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x') =_{M(k)} (i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x'))}$$

Therefore, the following delivation holds.

$$\dfrac{\begin{array}{llllll} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ M(k) : \mathbf{type} & (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x') : M(k) & (i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x')) : M(k) & \mathcal{T}_{14} & \mathbf{p}_{13}(\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x')) : (\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x') =_{M(k)} (i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x')) & \mathcal{D}_{2\text{–}3\text{–}1(1)} \end{array} \quad u_3(x') : \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f\big((\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x')\big) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f\big((\pi_1(\mathbf{i}(k)(d)(\mathbf{p}_{19}(u'))))(x')\big) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right]}{\mathbf{p}_{15}(\mathbf{p}_{13}(\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x')); u_3(x')) : \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f((i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x'))) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f((i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x'))) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right]}\,(\text{Th15})$$

$\mathcal{T}_{14}$:

$$\lambda y. \left[\begin{array}{l} \left(v_1 : \left[\begin{array}{l} z : A \\ f(y) =_{A\uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(m) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array}\right] \\ \left(v_2 : \left[\begin{array}{l} z : B \\ f(y) =_{A\uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : M(n) \\ h(x) =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array}\right] \end{array}\right] : M(k) \to \mathbf{type}$$

Let

$$\mathbf{v}_1 \overset{def}{\equiv} \pi_1(\mathbf{p}_{15}(\mathbf{p}_{13}(\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x')); u_3(x')))$$

$$\mathbf{v}_2 \overset{def}{\equiv} \pi_2(\mathbf{p}_{15}(\mathbf{p}_{13}(\mathbf{p}_{27}(k)(d)(\pi_1(u'))(\pi_2(u'))(x')); u_3(x')))$$

in

$$\mathbf{v}_1 : \left(v_1 : \left[\begin{array}{l} z : A \\ f\left((i_{\mathsf{suc}(d)\leq k})(\mathsf{inl}(x'))\right) =_{A\uplus B} \mathsf{inl}(z) \end{array} \right] \right) \to \left[\begin{array}{l} w : M(m) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inl}(w) \end{array} \right] \tag{5.120}$$

$$\mathbf{v}_2 : \left(v_2 : \left[\begin{array}{l} z : B \\ f\left((i_{\mathsf{suc}(d)\leq k})(\mathsf{inl}(x'))\right) =_{A\uplus B} \mathsf{inr}(z) \end{array} \right] \right) \to \left[\begin{array}{l} w : M(n) \\ h(x') =_{M(m)\uplus M(n)} \mathsf{inr}(w) \end{array} \right]. \tag{5.121}$$

Now assume

$$v_1' : \left[\begin{array}{l} z : A \\ f\left((i_{\mathsf{suc}(d)\leq k})(\mathsf{inl}(x'))\right) =_{A\uplus B} \mathsf{inl}(z) \end{array} \right] \tag{5.122}$$

By applying ΠE with (5.120) and (5.122) and then applying ΣE_R, we obtain:

$$\pi_2(\mathbf{v}_1(v_1')) : \ h(x') =_{M(m)\uplus M(n)} \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1'))) \tag{5.123}$$

By $=I$,

$$\begin{array}{l} \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z))) : \\ \quad \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \end{array} \tag{5.124}$$

By (5.124) and substitution by identity (Theorem 15) with (5.123), we have

$$\begin{array}{l} \mathbf{p}_{15}(\pi_2(\mathbf{v}_1(v_1')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) : \\ \quad \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \qquad =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{case}\ \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1')))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)), \end{array}$$

from which we can obtain

$$\begin{array}{l} \mathbf{p}_{15}(\pi_2(\mathbf{v}_1(v_1')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) : \\ \quad h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(\mathsf{inl}(\pi_1(\mathbf{v}_1(v_1')))) \end{array} \tag{5.125}$$

by β-reduction and $conv$, as the following equivalence holds.

$$\begin{array}{rcl} \mathsf{case}\ \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1')))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) & \equiv_\beta & \mathsf{inl}(\mathsf{inl}(\pi_1(\mathbf{v}_1(v_1')))) \\ \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) & \equiv_\beta & h'(\mathsf{inl}(x')) \end{array}$$

By applying ΣI with $\mathsf{inl}(\pi_1(\mathbf{v}_1(v_1')))$: $M(\mathsf{suc}(m))$ and (5.125), we have

$$\langle \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1'))), \mathbf{p}_{15}(\pi_2(\mathbf{v}_1(v_1')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) \rangle : \begin{bmatrix} w : M(\mathsf{suc}(m)) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{bmatrix} \tag{5.126}$$

Thus, by ΠI with (5.122) and (5.126), we obtain

$$\lambda v_1'.\langle \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1'))), \mathbf{p}_{15}(\pi_2(\mathbf{v}_1(v_1')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) \rangle : \left(v_1' : \begin{bmatrix} z : A \\ f((i_{\mathsf{suc}(d)\leq k})(\mathsf{inl}(x'))) =_{A\uplus B} \mathsf{inl}(z) \end{bmatrix} \right) \to \begin{bmatrix} w : M(\mathsf{suc}(m)) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{bmatrix} \tag{5.127}$$

On the other hand, let us assume

$$v_2' : \begin{bmatrix} z : B \\ f(i_{\mathsf{suc}(d)\leq k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{bmatrix} \tag{5.128}$$

By applying ΠE with (5.121) and (5.128) and then applying ΣE_R, we obtain:

$$\pi_2(\mathbf{v}_2(v_2')) : h(x') =_{M(m)\uplus M(n)} \mathsf{inr}(\pi_1(\mathbf{v}_2(v_2'))) \tag{5.129}$$

Again, by $=I$,

$$\mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z))) : \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \tag{5.130}$$

By (5.130) and substitution by identity (15) with (5.129), we have

$$\mathbf{p}_{15}(\pi_2(\mathbf{v}_2(v_2')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) : \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{case}\ \mathsf{inr}(\pi_1(\mathbf{v}_2(v_2')))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)),$$

from which we can obtain

$$\mathbf{p}_{15}(\pi_2(\mathbf{v}_2(v_2')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))) : h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(\pi_1(\mathbf{v}_2(v_2'))) \tag{5.131}$$

by β-reduction and $conv$, as the following equivalence holds.

$$\begin{aligned}\mathsf{case}\ \mathsf{inr}(\pi_1(\mathbf{v}_2(v_2')))\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) &\equiv_\beta \mathsf{inr}(\pi_1(\mathbf{v}_2(v_2')))\\ \mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) &\equiv_\beta h'(\mathsf{inl}(x'))\end{aligned}$$

By applying ΣI with $\pi_1(\mathbf{v}_2(v_2')) : M(n)$ and (5.131), we have

$$\langle \pi_1(\mathbf{v}_2(v_2')), \mathbf{p}_{15}(\pi_2(\mathbf{v}_2(v_2')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))\rangle : \begin{bmatrix} w : M(n) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{bmatrix} \tag{5.132}$$

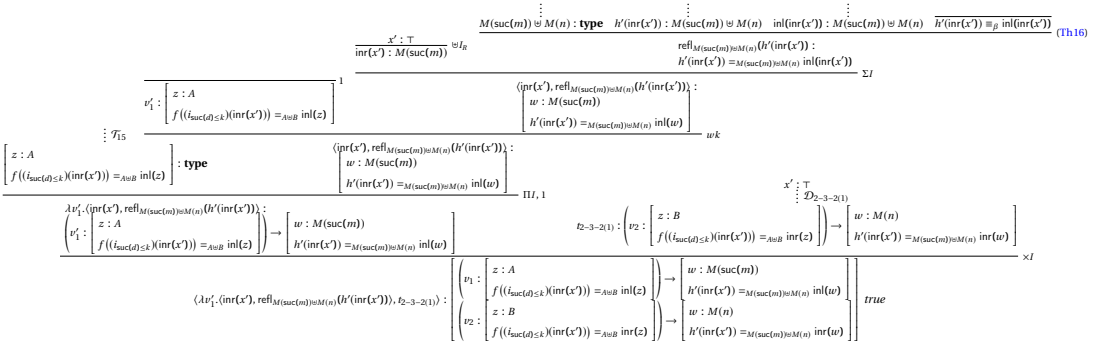
Thus, by ΠI with (5.128) and (5.132), we obtain

$$\lambda v_2'.\langle \pi_1(\mathbf{v}_2(v_2')), \mathbf{p}_{15}(\pi_2(\mathbf{v}_2(v_2')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))\rangle : \left(v_2' : \begin{bmatrix} z : B \\ f(i_{\mathsf{suc}(d)\le k}(x)) =_{A\uplus B} \mathsf{inr}(z) \end{bmatrix}\right) \to \begin{bmatrix} w : M(n) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{bmatrix} \tag{5.133}$$

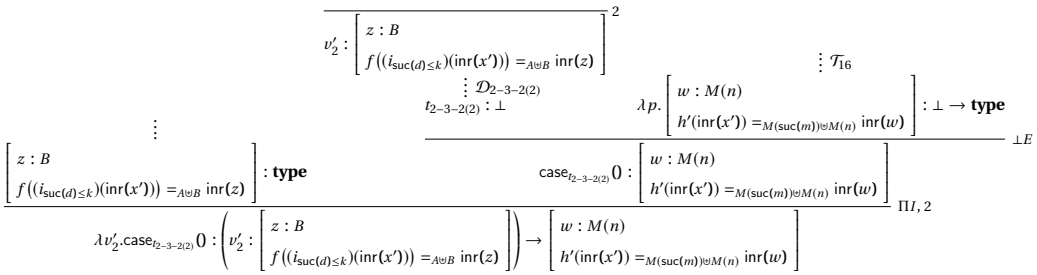
Thus, by applying ΣI with (5.127) and (5.133), we finally obtain the following proof as $t_{2\text{–}3\text{–}1}$.

$$\begin{aligned}&\langle \lambda v_1'.\langle \mathsf{inl}(\pi_1(\mathbf{v}_1(v_1'))), \mathbf{p}_{15}(\pi_2(\mathbf{v}_1(v_1')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))\rangle,\\ &\lambda v_2'.\langle \pi_1(\mathbf{v}_2(v_2')), \mathbf{p}_{15}(\pi_2(\mathbf{v}_2(v_2')); \mathsf{refl}_{M(\mathsf{suc}(m))\uplus M(n)}(\mathsf{case}\ h(x')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)))\rangle\rangle :\\ &\begin{bmatrix} \left(v_1 : \begin{bmatrix} z : A \\ f\left((i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x'))\right) =_{A\uplus B} \mathsf{inl}(z) \end{bmatrix}\right) \to \begin{bmatrix} w : M(\mathsf{suc}(m)) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inl}(w) \end{bmatrix} \\ \left(v_2 : \begin{bmatrix} z : B \\ f\left((i_{\mathsf{suc}(d)\le k})(\mathsf{inl}(x'))\right) =_{A\uplus B} \mathsf{inr}(z) \end{bmatrix}\right) \to \begin{bmatrix} w : M(n) \\ h'(\mathsf{inl}(x')) =_{M(\mathsf{suc}(m))\uplus M(n)} \mathsf{inr}(w) \end{bmatrix} \end{bmatrix}\end{aligned} \tag{5.134}$$

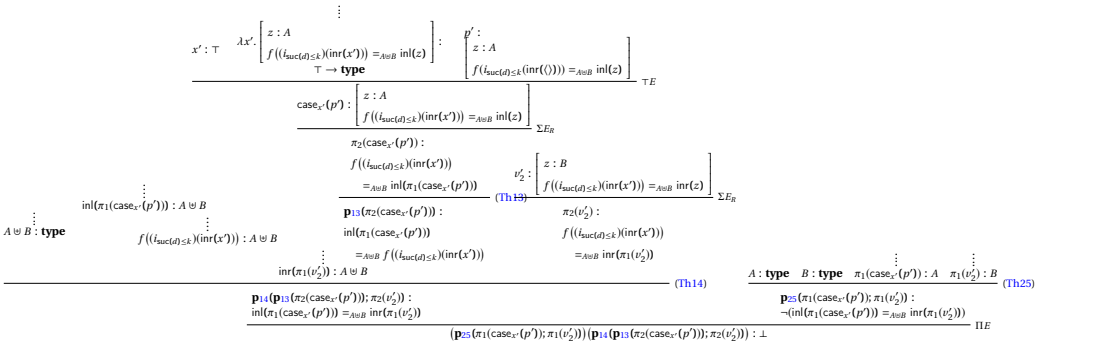
$\underline{\mathcal{D}_{2\text{–}3\text{–}2}}$:



$\underline{\mathcal{D}_{2\text{–}3\text{–}2(1)}}$:



$\underline{\mathcal{D}_{2\text{–}3\text{–}2(2)}}$:



Now we have proved $\mathcal{D}_{2\text{–}1}$, $\mathcal{D}_{2\text{–}2}$, and $\mathcal{D}_{2\text{–}3}$. $\mathcal{D}_2$ is thus given as in the derivation shown at (5.83).

$\underline{\mathcal{D}_3}$: Using the same technique as in the derivation of $\mathcal{D}_2$, we can obtain a counterpart of the proof t_2 from the assumption.

End of proof.

By Lemma 8 proved above, we can deduce the following corollary by putting $k' = k$.

Corollary 9:

$$(A, B : \mathbf{type}) \to (k : \mathbf{Nat}) \to (f : \mathbf{M}_k \to A \uplus B) \to \mathbf{bijection}\,(f) \to \left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} k = m + n \\ \left[\begin{array}{l} h : \mathbf{M}_k \to \mathbf{M}_m \uplus \mathbf{M}_n \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : \mathbf{M}_k) \to \left[\begin{array}{l} \left(u_3 : \left[\begin{array}{l} z : A \\ f(x) =_{A \uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_m \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(w) \end{array}\right] \\ \left(u_4 : \left[\begin{array}{l} z : B \\ f(x) =_{A \uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_n \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

5.3.6 Theorem: Finiteness of a part

Now we will prove the following theorem, which we mentioned in the beginning of the previous section.

Theorem 10

Given A : **type**, B : **type**, and k : **Nat**, if there exists a function f : $\mathbf{M}_k \to A \uplus B$ such that **bijection** (f) is true, there exists a function $g : \mathbf{M}_m \to A$ for some $m \leq k$ such that **bijection** (g) is true.

Proof. The proof will be divided into two steps: (1) first, we begin by constucting a function $\mathbf{M}_m \to A$ for some m : **Nat**; (2) next, we prove that the function is bijection.

(1) By assumption A : **type**, B : **type**, k : **Nat**, f : $\mathbf{M}_k \to A \uplus B$, and b_f : **bijection** (f), Corollary 9 yields a proof term of the following type. Let us write this term c.

$$\left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} k = m + n \\ \left[\begin{array}{l} h : \mathbf{M}_k \to \mathbf{M}_m \uplus \mathbf{M}_n \\ \left[\begin{array}{l} \mathbf{bijection}\,(h) \\ (x : \mathbf{M}_k) \to \left[\begin{array}{l} \left(u_3 : \left[\begin{array}{l} z : A \\ f(x) =_{A \uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_m \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(w) \end{array}\right] \\ \left(u_4 : \left[\begin{array}{l} z : B \\ f(x) =_{A \uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_n \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(w) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]$$

By applying ΣE several times to c, we get the following terms, which we abbreviate to m, n, h, b_h, and C, respectively.

$$\begin{array}{lll} m & \stackrel{def}{\equiv} & \pi_1(c) : \mathbf{Nat} \\ n & \stackrel{def}{\equiv} & \pi_1\pi_2(c) : \mathbf{Nat} \\ h & \stackrel{def}{\equiv} & \pi_1\pi_2\pi_2\pi_2(c) : \mathbf{M}_k \to \mathbf{M}_m \uplus \mathbf{M}_n \\ b_h & \stackrel{def}{\equiv} & \pi_1\pi_2\pi_2\pi_2\pi_2(c) : \mathbf{bijection}\,(h) \\ C & \stackrel{def}{\equiv} & \pi_2\pi_2\pi_2\pi_2\pi_2(c) : \\ & & (x : \mathbf{M}_k) \to \left[\begin{array}{l} \left(u_3 : \left[\begin{array}{l} z : A \\ f(x) =_{A \uplus B} \mathsf{inl}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_m \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(w) \end{array}\right] \\ \left(u_4 : \left[\begin{array}{l} z : B \\ f(x) =_{A \uplus B} \mathsf{inr}(z) \end{array}\right]\right) \to \left[\begin{array}{l} w : \mathbf{M}_n \\ h(x) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(w) \end{array}\right] \end{array}\right]. \end{array}$$

Term h^{-1} of type $\mathbf{M}_m \uplus \mathbf{M}_n \to \mathbf{M}_k$ stands for $\lambda w.\pi_1\pi_2 b_h(w)$, which is the inverse of h obtained from **surjection** (h) by the following derivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{b_h : \mathbf{bijection}\,(h)}{b_h : \left[\begin{array}{l}\mathbf{injection}\,(h)\\ \mathbf{surjection}\,(h)\end{array}\right]}\,def}{\pi_2 b_h : \mathbf{surjection}\,(h)}\times E \qquad \dfrac{\overline{w : \mathbf{M}_m \uplus \mathbf{M}_n}\,1}{\pi_2 b_h : (w : \mathbf{M}_m \uplus \mathbf{M}_n) \to \left[\begin{array}{l} y : \mathbf{M}_k \\ w =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(y)\end{array}\right]}\,def}{\dfrac{\pi_2 b_h(w) : \left[\begin{array}{l} y : \mathbf{M}_k \\ w =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(y)\end{array}\right]}{\pi_1\pi_2 b_h(w) : \mathbf{M}_k}\,\Sigma E}\,\Pi E}{\lambda w.\pi_1\pi_2 b_h(w) : \mathbf{M}_m \uplus \mathbf{M}_n \to \mathbf{M}_k}\,\Pi I, 1$$

Similarly, we obtain the inverse of f, $\lambda w.\pi_1\pi_2 b_f(w) : A \uplus B \to \mathbf{M}_k$, from $b_f : \mathbf{bijection}\,(f)$. We write it f^{-1}.

We construct $g' : (w : \mathbf{M}_m) \to \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w)))\end{array}\right]$ for $m : \mathbf{Nat}$, which gives $\lambda w.\pi_1 g'(w) : \mathbf{M}_m \to A$.

$$\dfrac{\mathbf{M}_m : \mathbf{type} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{h^{-1} : \mathbf{M}_m \uplus \mathbf{M}_n \to \mathbf{M}_k}{f : \mathbf{M}_k \to A \uplus B}\quad \dfrac{\dfrac{\overline{w : \mathbf{M}_m}^{\,4} \quad \mathbf{M}_n : \mathbf{type}}{\mathsf{inl}(w) : \mathbf{M}_m \uplus \mathbf{M}_n}\uplus I_L}{h^{-1}(\mathsf{inl}(w)) : \mathbf{M}_k}\to E}{f(h^{-1}(\mathsf{inl}(w))) : A \uplus B}\to E \qquad \lambda x.f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} x \to \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} x\end{array}\right] : A \uplus B \to \mathbf{type}}{\mathsf{case}\, f(h^{-1}(\mathsf{inl}(w)))\ \mathsf{of}\ (\lambda t.\lambda u.\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(t))\rangle, \lambda t.\lambda u.d_B) : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w))) \to \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w)))\end{array}\right]}\uplus E, 3 \qquad \dfrac{A \uplus B : \mathbf{type} \quad f(h^{-1}(\mathsf{inl}(w))) : A \uplus B}{\mathsf{refl}_{A\uplus B}(f(h^{-1}(\mathsf{inl}(w)))) : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w)))}=I}{(\mathsf{case}\, f(h^{-1}(\mathsf{inl}(w)))\ \mathsf{of}\ (\lambda t.\lambda u.\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(t))\rangle, \lambda t.\lambda u.d_B))(\mathsf{refl}_{A\uplus B}(f(h^{-1}(\mathsf{inl}(w))))) : \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w)))\end{array}\right]}\to E}{\lambda w.(\mathsf{case}\, f(h^{-1}(\mathsf{inl}(w)))\ \mathsf{of}\ (\lambda t.\lambda u.\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(t))\rangle, \lambda t.\lambda u.d_B))(\mathsf{refl}_{A\uplus B}(f(h^{-1}(\mathsf{inl}(w))))) : (w : \mathbf{M}_m) \to \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w)))\end{array}\right]}\Pi I, 4$$

$$\overline{t : A}^{\,3} \quad \overline{u : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} \mathsf{inl}(t)}^{\,1} \quad \vdots\ \mathcal{D}_A \qquad \langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(t))\rangle : \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} \mathsf{inl}(t)\end{array}\right]$$

$$\overline{t : B}^{\,3} \quad \overline{u : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} \mathsf{inr}(t)}^{\,2} \quad \vdots\ \mathcal{D}_B \qquad d_B : \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A\uplus B} \mathsf{inr}(t)\end{array}\right]$$

$\mathcal{D}_A$ and $\mathcal{D}_B$, as well as d_B, are given below.

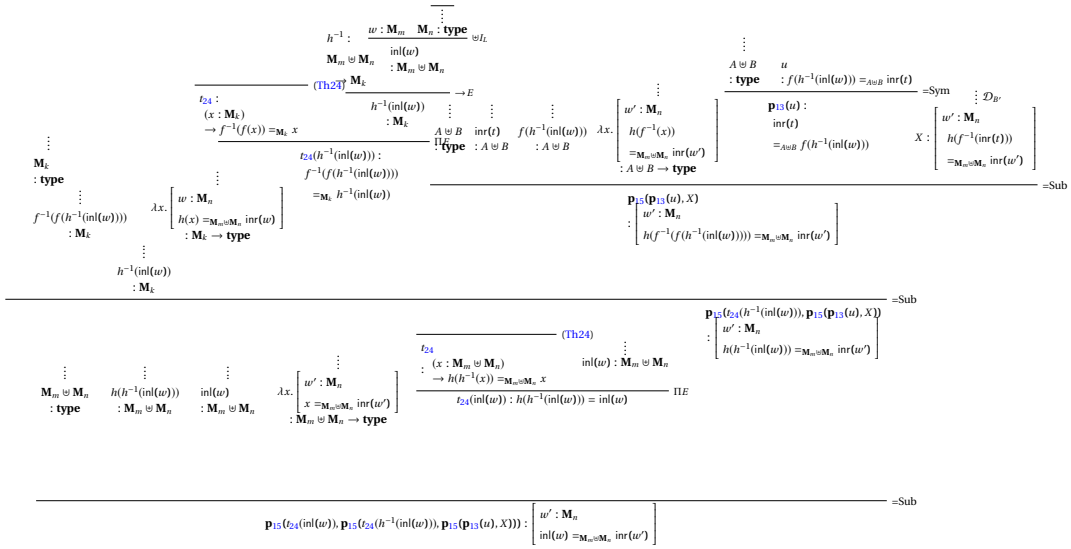
$\underline{\mathcal{D}_A}$

$$\dfrac{t : A \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdots}}{A \uplus B : \mathbf{type}} \qquad \dfrac{t : A}{\mathsf{inl}(t) : A \uplus B}\,\uplus I}{\mathsf{refl}_{A\uplus B}(\mathsf{inl}(t)) : \mathsf{inl}(t) =_{A\uplus B} \mathsf{inl}(t)}\,=I \qquad u : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} \mathsf{inl}(t)}{\mathsf{refl}_{A\uplus B}(\mathsf{inl}(t)) : \mathsf{inl}(t) =_{A\uplus B} \mathsf{inl}(t)}\,wk}{\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inl}(t))\rangle : \begin{bmatrix} a : A \\ \mathsf{inl}(a) =_{A\uplus B} \mathsf{inl}(t) \end{bmatrix}}\,\Sigma I$$

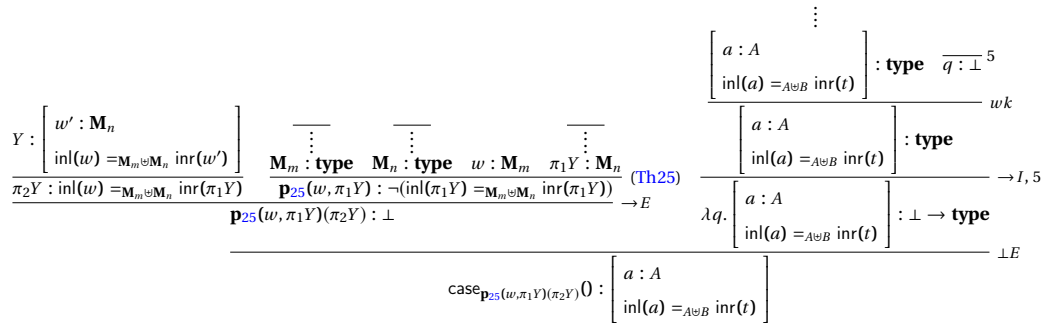
$\underline{\mathcal{D}_B}$ As the condition C stated above holds for f and h, term $f(h^{-1}(\mathsf{inl}(w))) : A \uplus B$ cannot consist of a proof of B. We will show that the assumption $t : B$ leads to a contradiction.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdots}}{A\uplus B : \mathbf{type}} \quad \dfrac{\vdots}{f(f^{-1}(\mathsf{inr}(t))) : A\uplus B} \quad \dfrac{\vdots}{\mathsf{inr}(t) : A\uplus B} \quad \dfrac{\vdots}{\lambda x.\left(u_4 : \begin{bmatrix} z : B \\ x =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix} : A\uplus B \to \mathbf{type}} \quad \dfrac{\dfrac{}{t_{24} : (x : A\uplus B) \to x =_{A\uplus B} f(f^{-1}(x))}\,(\text{Th24}) \quad \dfrac{\vdots}{\mathsf{inr}(t) : A\uplus B}}{t_{24}(\mathsf{inr}(t)) : f(f^{-1}(\mathsf{inr}(t))) =_{A\uplus B} \mathsf{inr}(t)}\,\Pi E \quad \dfrac{\dfrac{C : (x : \mathbf{M}_k) \to \begin{bmatrix} \left(u_3 : \begin{bmatrix} z : A \\ f(x) =_{A\uplus B} \mathsf{inl}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_m \\ h(x) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inl}(w)\end{bmatrix} \\ \left(u_4 : \begin{bmatrix} z : B \\ f(x) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_n \\ h(x) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix}\end{bmatrix} \qquad \dfrac{\dfrac{\dfrac{}{A : \mathbf{type}}\,con \quad t : B}{\mathsf{inr}(t) : A\uplus B}\,\uplus I \qquad f^{-1} : A\uplus B \to \mathbf{M}_k}{f^{-1}(\mathsf{inr}(t)) : \mathbf{M}_k}\,\to E}{C(f^{-1}(\mathsf{inr}(t))) : \begin{bmatrix} \left(u_3 : \begin{bmatrix} z : A \\ f(f^{-1}(\mathsf{inr}(t))) =_{A\uplus B} \mathsf{inl}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_m \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inl}(w)\end{bmatrix} \\ \left(u_4 : \begin{bmatrix} z : B \\ f(f^{-1}(\mathsf{inr}(t))) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix}\end{bmatrix}}\,\Pi E}{\pi_2 C(f^{-1}(\mathsf{inr}(t))) : \left(u_4 : \begin{bmatrix} z : B \\ f(f^{-1}(\mathsf{inr}(t))) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix}}\,\times E_L}{\mathbf{p}_{15}(t_{24}(\mathsf{inr}(t)), \pi_2 C(f^{-1}(\mathsf{inr}(t)))) : \left(u_4 : \begin{bmatrix} z : B \\ \mathsf{inr}(t) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}\right) \to \begin{bmatrix} w : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix}}\,{=}\text{Sub} \qquad \dfrac{t : B \qquad \dfrac{\dfrac{\dfrac{}{\vdots}}{A\uplus B : \mathbf{type}} \quad \dfrac{\dfrac{}{A : \mathbf{type}}\,con \quad t : B}{\mathsf{inr}(t) : A\uplus B}\,\uplus I_R}{\mathsf{refl}_{A\uplus B}(\mathsf{inr}(t)) : \mathsf{inr}(t) =_{A\uplus B} \mathsf{inr}(t)}\,=I}{\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inr}(t))\rangle : \begin{bmatrix} z : B \\ \mathsf{inr}(t) =_{A\uplus B} \mathsf{inr}(z)\end{bmatrix}}\,\Sigma I}{\mathbf{p}_{15}(t_{24}(\mathsf{inr}(t)), \pi_2 C(f^{-1}(\mathsf{inr}(t))))(\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inr}(t))\rangle) : \begin{bmatrix} w : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w)\end{bmatrix}}\,\Pi E$$

Let X stand for $\mathbf{p}_{15}(t_{24}(\mathsf{inr}(t)), \pi_2 C(f^{-1}(\mathsf{inr}(t))))(\langle t, \mathsf{refl}_{A\uplus B}(\mathsf{inr}(t))\rangle)$, the proof term constructed by the derivation above. The term X has type $\begin{bmatrix} w' : \mathbf{M}_n \\ h(f^{-1}(\mathsf{inr}(t))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w') \end{bmatrix}$. As we have assumed $u : f(h^{-1}(\mathsf{inl}(w))) =_{A\uplus B} \mathsf{inr}(t)$, we can have a proof of $\begin{bmatrix} w' : \mathbf{M}_n \\ h(f^{-1}(f(h^{-1}(\mathsf{inl}(w))))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w') \end{bmatrix}$. As f^{-1} and h^{-1} are inverse of f and h, respectively, we will get a proof of $\begin{bmatrix} w' : \mathbf{M}_n \\ \mathsf{inl}(w) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inr}(w') \end{bmatrix}$. We show the derivation below.



Now we have $\mathbf{p}_{15}(t_{24}(\mathsf{inl}(w)), \mathbf{p}_{15}(t_{24}(h^{-1}(\mathsf{inl}(w))), \mathbf{p}_{15}(\mathbf{p}_{13}(u), X)))$ of type $\left[\begin{array}{l} w' : \mathbf{M}_n \\ \mathsf{inl}(w) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(w') \end{array}\right]$. We write it Y. The following derivation shows that we can get a proof of $\left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A \uplus B} \mathsf{inr}(t) \end{array}\right]$, as $\pi_2 Y : \mathsf{inl}(w) =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(\pi_1 Y)$ leads to a contradiction with Theorem 25.

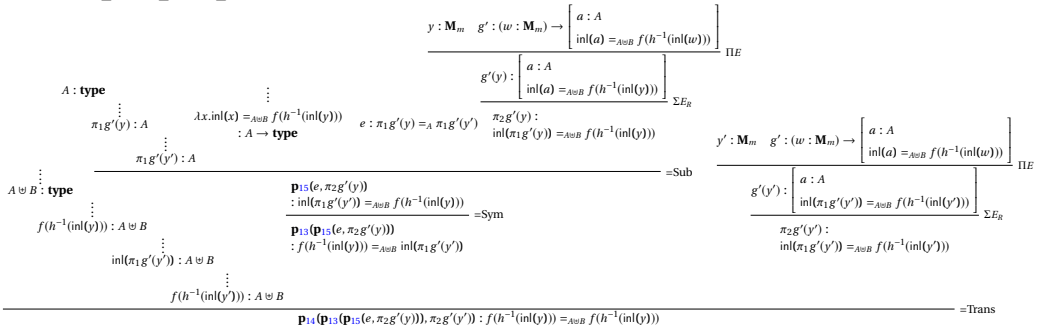


We have constructed a proof of $(w : \mathbf{M}_m) \to \left[\begin{array}{l} a : A \\ \mathsf{inl}(a) =_{A \uplus B} f(h^{-1}(\mathsf{inl}(w))) \end{array}\right]$, which we write g'. This yields $\lambda w.\pi_1 g'(w) : \mathbf{M}_m \to A$, where $m \leq k$.

(2) We next prove that **bijection** $(\lambda w.\pi_1 g'(w))$ holds, namely, (i) **injection** $(\lambda w.\pi_1 g'(w))$ and (ii) **surjection** $(\lambda w.\pi_1 g'(w))$ are true.

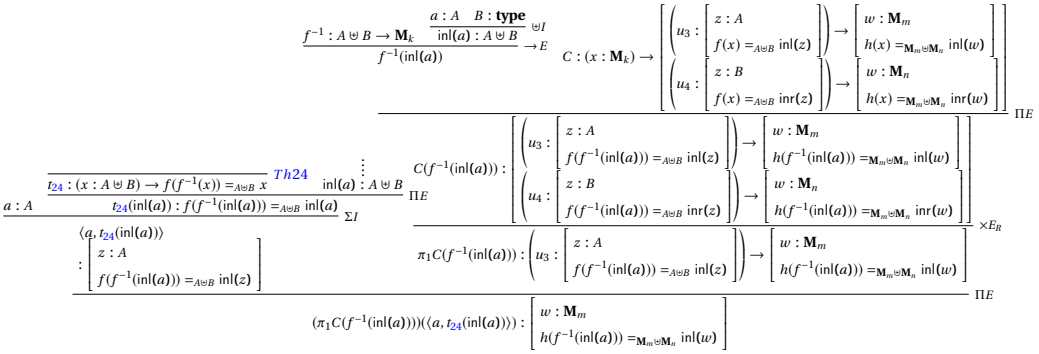
(i) **injection** $(\lambda w.\pi_1 g'(w))$ We prove $y =_{\mathbf{M}_k} y'$ under the assumption $y : \mathbf{M}_m$, $y' : \mathbf{M}_m$, and $e : \pi_1 g'(y) =_A \pi_1 g'(y')$. By the following derivation, we

obtain $\mathbf{p}_{14}(\mathbf{p}_{13}(\mathbf{p}_{15}(e,\pi_2 g'(y))),\pi_2 g'(y')) : f(h^{-1}(\mathsf{inl}(y))) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(y)))$.



Now, **injection** (f) holds according to the assumption. As **bijection** (h) holds, **injection** $\left(h^{-1}\right)$ holds by Theorem 24. By Lemma 20, **injection** $(\lambda x.\mathsf{inl}(x))$ also holds. Hence, by Lemma 22, it follows that **injection** $\left(\lambda x.f(h^{-1}(\mathsf{inl}(x)))\right)$. Therefore, $y =_{\mathbf{M}_m} y'$.

(ii) **surjection** $(\lambda w.\pi_1 g'(w))$ We prove $\left[\begin{array}{l} w : \mathbf{M}_m \\ a =_A \pi_1 g'(w) \end{array}\right]$ by assuming $a : A$.



Let Z stand for the constructed term,

$$(\pi_1 C(f^{-1}(\mathsf{inl}(a))))(\langle a, t_{24}(\mathsf{inl}(a))\rangle) : \left[\begin{array}{l} w : \mathbf{M}_m \\ h(f^{-1}(\mathsf{inl}(a))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inl}(w) \end{array}\right].$$

From this, We have

$$\pi_1 Z : \mathbf{M}_m \tag{5.135}$$

$$\pi_2 Z : h(f^{-1}(\mathsf{inl}(a))) =_{\mathbf{M}_m\uplus\mathbf{M}_n} \mathsf{inl}(\pi_1 Z). \tag{5.136}$$

Now we prove $\mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z)))$ and $f(h^{-1}(\mathsf{inl}(\pi_1 Z))) =_{A\uplus B} \mathsf{inl}(\pi_1 g'(\pi_1 Z))$.

First we prove $\mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z)))$. By $=I$,

$$\mathsf{refl}_{A\uplus B}(f(h^{-1}(h(f^{-1}(\mathsf{inl}(a)))))) : f(h^{-1}(h(f^{-1}(\mathsf{inl}(a))))) =_{A\uplus B} f(h^{-1}(h(f^{-1}(\mathsf{inl}(a)))))$$

By =Sub with (5.136),

$$\begin{aligned}&\mathbf{p}_{15}(\pi_2 Z, \mathsf{refl}_{A\uplus B}(f(h^{-1}(h(f^{-1}(\mathsf{inl}(a))))))) \\ &: f(h^{-1}(h(f^{-1}(\mathsf{inl}(a))))) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z)))\end{aligned} \tag{5.137}$$

As we have $t_{24} : (x : \mathbf{M}_k) \to h^{-1}(h(x)) =_{\mathbf{M}_k} x$ (Theorem 24),

$$t_{24}(f^{-1}(\mathsf{inl}(a))) : h^{-1}(h(f^{-1}(\mathsf{inl}(a)))) =_{\mathbf{M}_k} f^{-1}(\mathsf{inl}(a))$$

holds for $f^{-1}(\mathsf{inl}(a))$. Hence, by =Sub with (5.137),

$$\begin{aligned}&\mathbf{p}_{15}(t_{24}(f^{-1}(\mathsf{inl}(a))), \mathbf{p}_{15}(\pi_2 Z, \mathsf{refl}_{A\uplus B}(f(h^{-1}(h(f^{-1}(\mathsf{inl}(a)))))))) \\ &: f(f^{-1}(\mathsf{inl}(a))) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z)))\end{aligned} \tag{5.138}$$

Similarly, as we have $t_{24} : (x : A \uplus B) \to f(f^{-1}(x)) =_{A\uplus B} x$ (Theorem 24),

$$t_{24}(\mathsf{inl}(a)) : f(f^{-1}(\mathsf{inl}(a))) =_{A\uplus B} \mathsf{inl}(a)$$

holds for $\mathsf{inl}(a)$. Hnce, by =Sub with (5.138),

$$\begin{aligned}&\mathbf{p}_{15}(t_{24}(\mathsf{inl}(a)), \mathbf{p}_{15}(t_{24}(f^{-1}(\mathsf{inl}(a))), \mathbf{p}_{15}(\pi_2 Z, \mathsf{refl}_{A\uplus B}(f(h^{-1}(h(f^{-1}(\mathsf{inl}(a)))))))))) \\ &: \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z))).\end{aligned} \tag{5.139}$$

Next, we prove $f(h^{-1}(\mathsf{inl}(\pi_1 Z))) =_{A\uplus B} \mathsf{inl}(\pi_1 g'(\pi_1 Z))$. By applying ΠE with (5.135) and $g' : (w : \mathbf{M}_m) \to \begin{bmatrix} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(w))) \end{bmatrix}$,

$$g'(\pi_1 Z) : \begin{bmatrix} a : A \\ \mathsf{inl}(a) =_{A\uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z))) \end{bmatrix}.$$

By ΣE_L,

$$\pi_2 g'(\pi_1 Z) : \mathsf{inl}(\pi_1 g'(\pi_1 Z)) =_{A \uplus B} f(h^{-1}(\mathsf{inl}(\pi_1 Z)))$$

By =Sym,

$$\mathbf{p}_{13}(\pi_2 g'(\pi_1 Z)) : f(h^{-1}(\mathsf{inl}(\pi_1 Z))) =_{A \uplus B} \mathsf{inl}(\pi_1 g'(\pi_1 Z)) \tag{5.140}$$

By =Trans with (5.139) and (5.140),

$$\begin{aligned}\mathbf{p}_{14}(\mathbf{p}_{15}(t_{24}(\mathsf{inl}(a)), \mathbf{p}_{15}(t_{24}(f^{-1}(\mathsf{inl}(a))), \mathbf{p}_{15}(\pi_2 Z, \mathsf{refl}_{A \uplus B}(f(h^{-1}(h(f^{-1}(\mathsf{inl}(a))))))))), \\ \mathbf{p}_{13}(\pi_2 g'(\pi_1 Z))) \\ : \mathsf{inl}(a) =_{A \uplus B} \mathsf{inl}(\pi_1 g'(\pi_1 Z)).\end{aligned} \tag{5.141}$$

Let W stand for the proof term of (5.141). According to Lemma 20, we have

$$l_{20} : (A, B : \mathbf{type}) \to (y, y' : A) \to \mathsf{inl}(y) =_{A \uplus B} \mathsf{inl}(y') \to y =_A y'. \tag{5.142}$$

By the lemma and (5.141), we obtain

$$l_{20}(A)(B)(a)(\pi_1 g'(\pi_1 Z))(W) : a =_A \pi_1 g'(\pi_1 Z) \tag{5.143}$$

Thus, by applying ΣE with (5.135) and (5.143), we finally get

$$\langle \pi_1 Z, l_{20}(A)(B)(a)(\pi_1 g'(\pi_1 Z))(W) \rangle : \begin{bmatrix} w : \mathbf{M}_m \\ a =_A \pi_1 g'(w) \end{bmatrix}$$

End of proof.

We can rephrase Theorem 10 as follows.

$$\Gamma,\ A : \mathbf{type},\ B : \mathbf{type},\ a : \mathbf{Finite}\,(A \uplus B) \vdash \mathbf{Finite}\,(A)\ \ true$$

5.4 Right monotonicity

We show that, among type $Q^{\exists}$ quantifiers in Table 5.1, $Q^{\exists}_{most}$ and $Q^{\exists}_{atLeast}$ are right upward monotone.

Theorem 11: Right upward Monotonicity ($Q^{\exists}_{most}$ and $Q^{\exists}_{atLeast}$)

$$\Gamma,\ a : \mathbf{Finite}_{\pi_1}\left(\left(x : \mathbf{entity}\right) \times A(x)\right),\ p : \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n \geq \mathcal{N} \\ \left[\begin{array}{l} h : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(h) \end{array}\right] \end{array}\right] \end{array}\right],$$

$$t : \left(x : \mathbf{entity}\right) \rightarrow (u : A(x)) \rightarrow B(x) \rightarrow B'(x) \ \vdash \left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} m \geq \mathcal{N} \\ \left[\begin{array}{l} g : \mathbf{M}_m \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B'(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(g) \end{array}\right] \end{array}\right] \end{array}\right] \ true$$

holds, where $\mathcal{N}$ is a natural number and

$$\Gamma \ \vdash \ A : \mathbf{entity} \rightarrow \mathbf{type}$$

$$\Gamma, x : \mathbf{entity}, u : A(x) \ \vdash \ B : \mathbf{entity} \rightarrow \mathbf{type}$$

$$\Gamma, x : \mathbf{entity}, u : A(x) \ \vdash \ B' : \mathbf{entity} \rightarrow \mathbf{type}$$

Proof. The proof will be devided into two steps.

In the first step, we construct a number that corresponds to m in the right hand side. The number m is obtained by proving $\mathbf{Finite}_{\pi_1}\left(\left(x : \mathbf{entity}\right) \times (u : A(x)) \times B'(x)\right)$ from $a : \mathbf{Finite}_{\pi_1}\left(\left(x : \mathbf{entity}\right) \times A(x)\right)$. We then show that $m \geq \mathcal{N}$ holds in the second step.

For the preparation, we apply ΣE to a and p several times and obtain

the following proof terms. For abbreviation, we use k, r, f, and b_f.

$$\begin{aligned} k &\equiv \pi_1 a : \mathbf{Nat} \\ f &\equiv \pi_1\pi_2 a : \mathbf{M}_k \to \big(x : \mathbf{entity}\big) \times A(x) \\ b_f &\equiv \pi_2\pi_2 : \mathbf{bijection}_{\pi_1}(f) \end{aligned} \tag{5.144}$$

$$\begin{aligned} n &\equiv \pi_1 p : \mathbf{Nat} \\ r &\equiv \pi_1\pi_2 p : n \geq \mathcal{N} \\ h &\equiv \pi_1\pi_2\pi_2 p : \mathbf{M}_n \to \big(x : \mathbf{entity}\big) \times (u : A(x)) \times B(x) \\ b_h &\equiv \pi_2\pi_1\pi_2 p_1 : \mathbf{bijection}_{\pi_1}(h) \end{aligned} \tag{5.145}$$

(1) From f and b_f, we can obtain the following along the process we have shown in Figure 5.1.

$$m : \mathbf{Nat} \tag{5.146}$$

$$g : \mathbf{M}_m \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B'(x) \end{array}\right] \end{array}\right] \tag{5.147}$$

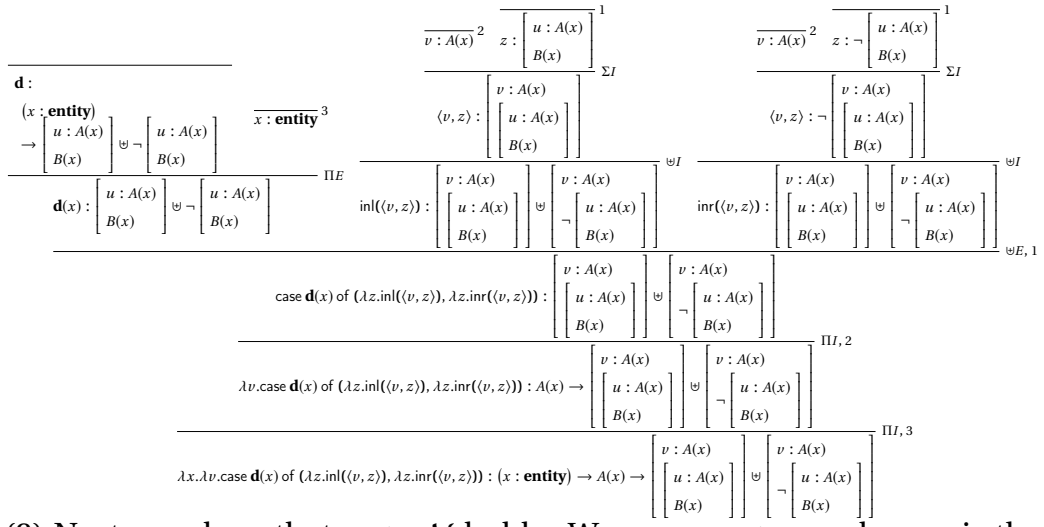
$$b_g : \mathbf{bijection}_{\pi_1}(g) \tag{5.148}$$

In the step $\langle 2 \rangle$, the following have to hold in order to apply Theorem 4.

$$\big(x : \mathbf{entity}\big) \to A(x) \to \left[\begin{array}{l} v : A(x) \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \uplus \left[\begin{array}{l} v : A(x) \\ \neg \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \tag{5.149}$$

$$\big(x : \mathbf{entity}\big) \to \left[\begin{array}{l} v : A(x) \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \uplus \left[\begin{array}{l} v : A(x) \\ \neg \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \to A(x) \tag{5.150}$$

(5.150) obviously holds. (5.149) is derived as follows, where **d** is a proof that $(u : A(x)) \times B(x)$ is a decidable predicate.



(2) Next, we show that $m \geq \mathcal{N}$ holds. We prove $m \geq n$, where n is the number that satisfy $n \geq \mathcal{N}$ given in the premise p. It suffices to show that there exists an injection from $\mathbf{M}_n$ to $\mathbf{M}_m$.[5]

First, we have $h : \mathbf{M}_n \to \left(x : \mathbf{entity}\right) \times (u : A(x)) \times B(x)$. We can also construct

$$\lambda z.(\pi_1 z, (\pi_1\pi_2 z, t(\pi_1 z)(\pi_1\pi_2 z)(\pi_2\pi_2 z))) : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B(x) \end{array}\right] \end{array}\right] \to \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B'(x) \end{array}\right] \end{array}\right]$$

with the second premise $t : \left(x : \mathbf{entity}\right) \to (u : A(x)) \to B(x) \to B'(x)$. We write the term q.

By applying ΣE_L to $b_g : \mathbf{bijection}_{\pi_1}(g)$, we obtain $\pi_2 b_g : \mathbf{surjection}_{\pi_1}(g)$, where $\mathbf{surjection}_{\pi_1}(g)$ is $\left(z : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} u : A(x) \\ B'(x) \end{array}\right] \end{array}\right]\right) \to \left[\begin{array}{l} y : \mathbf{M}_m \\ \pi_1 z =_{\mathbf{entity}} \pi_1 g(y) \end{array}\right]$ by

[5] We have provided the semantic representations based on the assumption that an injection corresponds to the small/large relation of numbers, which was also treated as given in Sundholm's original formulation. We assume that we can obtain the following theorem for free: For any natural number n and n', if there is an injection from $\mathbf{M}_n$ to $\mathbf{M}_{n'}$, then $n \leq n'$.

definition. We write this term s_g.

By combining h, q, and s_g, we can construct the following function.

$$\pi_1 \circ s_g \circ q \circ h : \mathbf{M}_n \to \mathbf{M}_m.$$

Next, we show that $\pi_1 \circ s_g \circ q \circ h : \mathbf{M}_n \to \mathbf{M}_m$ is an injection, i.e.,

$$(y : \mathbf{M}_n) \to (y' : \mathbf{M}_n) \to \pi_1 \circ s_g \circ q \circ h(y) =_{\mathbf{M}_m} \pi_1 \circ s_g \circ q \circ h(y') \to y =_{\mathbf{M}_n} y'.$$

By assuming $y : \mathbf{M}_n$, $y' : \mathbf{M}_n$, and $v : \pi_1 \circ s_g \circ q \circ h(y) =_{\mathbf{M}_m} \pi_1 \circ s_g \circ q \circ h(y')$, we prove $y =_{\mathbf{M}_n} y'$.

By sequentially applying ΠE with s_g, q, h, and y, we obtain

$$s_g(q(h(y))) : \left[\begin{array}{l} w : \mathbf{M}_m \\ \pi_1 q(h(y)) =_{\mathbf{entity}} \pi_1 g(w) \end{array} \right],$$

from which, by ΣE_L, we get

$$\pi_2 s_g(q(h(y))) : \pi_1 q(h(y)) =_{\mathbf{entity}} \pi_1 g(\pi_1(s_g(q(h(y))))). \tag{5.151}$$

Similarly, from s_g, q, h, and y', we obtain

$$\pi_2 s_g(q(h(y'))) : \pi_1 q(h(y')) =_{\mathbf{entity}} \pi_1 g(\pi_1(s_g(q(h(y'))))),$$

from which, by =Sym, we have

$$\mathbf{p}_{13}(\pi_2 s_g(q(h(y')))) : \pi_1 g(\pi_1(s_g(q(h(y'))))) =_{\mathbf{entity}} \pi_1 q(h(y')). \tag{5.152}$$

By applying =Sub with assumption $v : \pi_1 \circ s_g \circ q \circ h(y) =_{\mathbf{M}_m} \pi_1 \circ s_g \circ q \circ h(y')$ and (5.151), we get

$$\mathbf{p}_{15}(v, \pi_2 s_g(q(h(y)))) : \pi_1 q(h(y)) =_{\mathbf{entity}} \pi_1 g(\pi_1(s_g(q(h(y'))))). \tag{5.153}$$

By applying =Trans with (5.152) and (5.153), we get

$$\begin{aligned} &\mathbf{p}_{14}(\mathbf{p}_{15}(v, \pi_2 s_g(q(h(y)))), \mathbf{p}_{13}(\pi_2 s_g(q(h(y'))))) \\ &\quad : \pi_1 q(h(y)) =_{\mathbf{entity}} \pi_1 q(h(y')) \end{aligned} \tag{5.154}$$

As q is $\lambda z.(\pi_1 z, (\pi_1\pi_2 z, p_2(\pi_1 z)(\pi_2\pi_2 z)))$, $\lambda x.\pi_1 q(x) \equiv_\beta \lambda x.\pi_1(x)$. Therefore,

$$\mathbf{p}_{14}(\mathbf{p}_{15}(v, \pi_2 s_g(q(h(y)))), \mathbf{p}_{13}(\pi_2 s_g(q(h(y'))))) \\ : \pi_1(h(y)) =_{\mathbf{entity}} \pi_1(h(y')) \tag{5.155}$$

Now, as there exists proof i_h of $\mathbf{injection}_{\pi_1}(h)$, we obtain the following with $y : \mathbf{M}_n$ and $y' : \mathbf{M}_n$.

$$i_h(y)(y') : \pi_1(h(y)) =_{\mathbf{entity}} \pi_1(h(y')) \to y =_{\mathbf{M}_n} y' \tag{5.156}$$

Thus, by applying ΠE with (5.155) and (5.156), we finally get

$$i_h(y)(y')(\mathbf{p}_{14}(\mathbf{p}_{15}(v, \pi_2 s_g(q(h(y)))), \mathbf{p}_{13}(\pi_2 s_g(q(h(y')))))) \\ : y =_{\mathbf{M}_n} y' \tag{5.157}$$

Therefore, we conclude that there exists an injection $\mathbf{M}_n \to \mathbf{M}_m$.

End of proof.

Here is a remark on step ⟨2⟩, where we applied Theorem 4. In the above proof, we transformed $A(x)$ to $(v : A(x)) \times (u : A(x)) \times B(x) \uplus (v : A(x)) \times \neg(u : A(x)) \times B(x)$, which are mutually deducible. One might think that $A(x)$ to $(u : A(x)) \times B(x) \uplus (u : A(x)) \times \neg B(x)$ also works. Although $A(x)$ and $(u : A(x)) \times B(x) \uplus (u : A(x)) \times \neg B(x)$ are also mutually deducible, this is not the case because $(u : A(x)) \times B(x)$ and $(u : A(x)) \times \neg B(x)$ are not contradictory when u is appearing in B as a free variable: for $a : (u : A(x)) \times B(x)$ and $a' : (u : A(x)) \times \neg B(x)$, we only get $\pi_2 a : B_{[\pi_1 a/u]}$ and $\pi_2 a' : \neg B_{[\pi_1 a'/u]}$, which are apparently different propositions. In other words, when there is an anaphoric dependency between A and B, there may exist an entity x that satisfies $(u : A(x)) \times B(x)$ on one hand and satisfies $(u : A(x)) \times \neg B(x)$ on the other hand. Thus, we cannot apply Theorem 5 to split the entities into two groups, $\{x \mid (u : A(x)) \times B(x)\}$ and $\{x \mid (u : A(x)) \times \neg B(x)\}$.

We are familiar with such a situation in the interpretation of donkey sentences. We repeat an example below.

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

Recall that, since farmers can have multiple donkeys, there are two ways of counting that has been taken into account in the literature. One is to consider farmers that beat every donkey they own (*universal reading*, or *strong reading*). The other is to consider farmers that beat some donkey they own (*existential reading*, or *weak reading*). In our proof above, we split entities into two groups, where the first group, $\{x \mid (v : A(x)) \times (u : A(x)) \times B(x)\}$, consists of entities considered in the existential reading.

Right downward monotonicity of $Q^{\exists}_{atMost}$ can be proved in an analogous way: when $n \leq \mathcal{N}$, one can prove $m \leq \mathcal{N}$ from $m \leq n$, which is similarly proved by using the assumption $(x : \mathbf{entity}) \to (u : A(x)) \to B'(x) \to B(x)$.

5.5 Left monotonicity

It is, in fact, not possible to give proofs of left monotonicity following a similar strategy. In this section, we describe the problem.

We repeat the model-theoretic definition of left upward monotonicity below.

Quantifier Q is *Left upward monotone* ($\uparrow$MON) if and only if, for all M, $B \subseteq M$, and $A \subseteq A' \subseteq M$, $Q_M(A, B)$ implies $Q_M(A', B)$.

If we write this definition as a theorem of dependent type theory using the same idea, we get the following.

A quantifier **Q** is left upward monotone if the judgement

$$\Gamma,\ p : \mathbf{Q}(A, B),\ q : \big(x : \mathbf{entity}\big) \to (v : A(x)) \to A'(x) \vdash \mathbf{Q}(A', B)\ \mathit{true}$$

holds, where

$$\Gamma \vdash A : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity},\ u : A(x) \vdash B : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma, x : \mathbf{entity},\ u' : A'(x) \vdash B : \mathbf{entity} \to \mathbf{type}$$
$$\Gamma \vdash \mathbf{Q}(A, B) : \mathbf{type}$$

The inclusion relation between A and A' is again represented as a proposition about entities. Here, unlike right upward monotonicity, we do not have the elements to the left of A.

This formulation, however, works only when B on the left-hand side does not depend on the proof of A. That is, we can account for (5.4) but neither (5.5), (5.6), nor (5.7).

(5.4) Some young man who owns a donkey is happy.
⇒ Some man who owns a donkey is happy.

(5.5) Some young man who owns a^1 donkey loves it$_1$.
⇒ Some man who owns a^2 donkey loves it$_2$.

(5.6) Some man who owns a^1 female donkey loves it$_1$.
⇒ Some man who owns a^1 donkey loves it$_1$.

(5.7) Some man who owns a^1 garden sprinkles it$_1$.
⇏ Some man who owns a^1 house sprinkles it$_1$.

To make the illustration simpler, let us consider the classical *some* which is represented by Σ-type. Since the problem we encounter here comes

from the nature of dependent types, we can still observe the same problem in such a simplified case.

The entailment exemplified in (5.4) corresponds to the following inference, which is straightforwardly provable. In this case, the formulation we had above captures the left upward monotone inference.

$$\Gamma,\ p : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{ll} & z : \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{young}(x) \end{array}\right] \\ u : & \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{happy}(x) \end{array}\right],$$

$$q : (x : \mathbf{entity}) \to \left(\begin{array}{ll} & z : \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{young}(x) \end{array}\right] \\ u : & \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right) \to \left[\begin{array}{l} z : \mathbf{man}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right]$$

$$\vdash \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{ll} & z : \mathbf{man}(x) \\ u : & \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{happy}(x) \end{array}\right] \mathit{true}$$

However, it is not the case in (5.5), which contains the donkey anaphora. The following inference corresponding to (5.5) does not hold in general.

$$\Gamma,\ p : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \\ u : \left[\begin{array}{l} z : \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{young}(x) \end{array}\right] \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{love}(x, \pi_1\pi_2 u) \end{array}\right] \end{array}\right],$$

$$q : (x : \mathbf{entity}) \to \left(u : \left[\begin{array}{l} z : \left[\begin{array}{l} \mathbf{man}(x) \\ \mathbf{young}(x) \end{array}\right] \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right]\right) \to \left[\begin{array}{l} z : \mathbf{man}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right]$$

$$\vdash \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \\ u' : \left[\begin{array}{l} z : \mathbf{man}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{love}(x, \pi_1\pi_2 u') \end{array}\right] \end{array}\right] \mathit{true}$$

This is because the consequence of the assumption q introduces a new existentially-quantified entity, which is a donkey, for the given proof of *young man who owns a donkey*. It is not easy to prove that the newly introduced donkey also satisfies the predicate **love**.

In case of (5.5), since it is provable that young man is a man without any additional assumption, we can prove the entailment without using the assumption q. If we can ignore using the assumption q, it becomes easy to prove the left upward monotone inference in (5.5). The similar applies to (5.6).

This observation suggests that interpreting the inclusion $A \subseteq A'$ as $(x : \mathbf{entity}) \to (u : A(x)) \to A'(x)$ is not sufficient to account for left mono-

tone inference involving donkey anaphora. We need to take into account the inner structure of A and A', which are possibly referred in the nuclear scope B.

This point has been discussed in Kanazawa (1994). The study considers the difference in dynamic effects between *man who owns a garden* and *man who owns a house* in the framework of DPL. In the analysis, the issue was how to block the entailment such as (5.7) among the positive examples involving donkey anaphora. Here, however, it is the opposite: the naïve formalization blocks the entailment like (5.5), (5.6), and (5.7) involving donkey anaphora in principle, as the assumption introduces different proof terms. Thus, it is necessary to account for the case such as (5.5) and (5.6), where the entailment is valid. The analysis of left monotonicity in DTS and the comparison with DPL are left for our future work.

5.6 Anaphora and inference

We have proved conservativity and right monotonicity of quantifiers as theorems in dependent type theory. The theorems also take into account the cases where an anaphoric dependency is involved between restrictor and scope. By using those theorems, we can show that the following entailment holds in a proof-theoretic manner.

(5.8)

P_1	Most farmers who own a^1 donkey love it$_1$.
P_2	Every farmer who loves a donkey are kind.
H	Most farmers who own a donkey are kind.

We sketch how to prove the entailment relation.

Step 1. Obtaining semantic representations. We first obtain semantic representations of P_1, P_2, and H by semantic composition.

$$(\mathrm{P}_1)\quad \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n \geq \mathbf{majorityOf}\left(\pi_1\pi_1\left(@_2 :: \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right]\right)\right) \\ \left[\begin{array}{l} h : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} w : \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \\ \mathbf{love}(x, @_1 :: \mathbf{entity}) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(h) \end{array}\right] \end{array}\right] \end{array}\right]$$

$$(\mathrm{P}_2)\quad \left(z : \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{love}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \rightarrow \mathbf{kind}(\pi_1 z)$$

(H)
$$\left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} m \geq \mathbf{majorityOf}\left(\pi_1\pi_1\left(@_3 :: \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}\left(\left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]\right) \\ \pi_1(a) \geq 1 \end{array}\right]\right)\right) \\ \left[\begin{array}{l} g : \mathbf{M}_m \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{kind}(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(g) \end{array}\right] \end{array}\right] \end{array}\right]$$

For the quantifier *every* in (P_2), we adopt the universal quantifier in dependent type theory, Π-type, for simplicity. P_1 and H is underspecified semantic representation involving @-terms.

Step 2. Type checking, Proof search, and @-elimination. Next, anaphora resolution takes place. In order for the entailment relation in (5.8) to hold, we need to assume that P_1 and H consider the same domain of *farmers who own a donkey*. This is an admissible assumption in entailment relations in general, as we interpret the hypothesis under the condition of premises. Thus, $@_2$ in (P_1) and $@_3$ in (H) is resolved by the same proof term, say a^+ in the global context. Let us abbreviate $\pi_1\pi_1 a^+ : \mathbf{Nat}$ to a'. $@_1$ is replaced with $\pi_1\pi_2 w$ (see Section 4.3).

Step 3. Conservativity inference of *most*. After the anaphora resolution, we obtain the following fully-specified semantic representation of (P_1).

$$(\mathrm{P}_1)\quad \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n \geq \mathbf{majorityOf}(a') \\ \left[\begin{array}{l} h : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{love}(x, \pi_1\pi_2 w) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(h) \end{array}\right] \end{array}\right] \end{array}\right]$$

By Theorem 2 of conservativity inference, we obtain the following formula, where we highlight the additional conjunct.

$$(5.9)\quad \left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n \geq \mathbf{majorityOf}(a') \\ \left[\begin{array}{l} h : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} v : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{love}(x, \pi_1\pi_2 w) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(h) \end{array}\right] \end{array}\right] \end{array}\right]$$

This roughly corresponds to *Most farmers who own a donkey are farmers who own a*^{1} *donkey and love it*$_1$. Thus, the inference step can be expressed as follows in natural language.

(5.10) CONSERVATIVITY INFERENCE

P_1 Most farmers who own a^1 donkey love it$_1$.

Most farmers who own a donkey are farmers who own a^1 donkey and love it$_1$.

Step 4. General inference of conjunction. From (5.9), we can obtain the following formula.

(5.11)
$$\left[\begin{array}{l} n : \mathbf{Nat} \\ \left[\begin{array}{l} n \geq \mathbf{majorityOf}(a') \\ \left[\begin{array}{l} h : \mathbf{M}_n \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} v : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{love}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(h) \end{array}\right] \end{array}\right] \end{array}\right]$$

The predicate **own** is eliminated from the highlighted part. The proof term of (5.11) can be constructed from a proof of (5.9) straightforwardly because it only eliminates the conjunct.[6]

The formula (5.11) can be interpreted as *Most farmers who own a donkey are farmers who love a donkey*. Thus, the inference step is expressed as follows.

(5.12) GENERAL INFERENCE

Most farmers who own a donkey are farmers who own a^1 donkey and love it$_1$.

Most farmers who own a donkey are farmers who love a donkey.

Step 5. Monotonicity inference of *most*. With (5.11) and (5.6), by Theorem 11 we obtain the following formula.

[6] The proof is very similar to the right-to-left direction of Theorem 2.

(5.13)
$$\left[\begin{array}{l} m : \mathbf{Nat} \\ \left[\begin{array}{l} m \geq \mathbf{majorityOf}(a') \\ \left[\begin{array}{l} g : \mathbf{M}_m \rightarrow \left[\begin{array}{l} x : \mathbf{entity} \\ \left[\begin{array}{l} w : \left[\begin{array}{l} \mathbf{farmer}(x) \\ \left[\begin{array}{l} y : \mathbf{entity} \\ \left[\begin{array}{l} u : \mathbf{donkey}(y) \\ \mathbf{own}(x,y) \end{array}\right] \end{array}\right] \end{array}\right] \\ \mathbf{kind}(x) \end{array}\right] \end{array}\right] \\ \mathbf{bijection}_{\pi_1}(g) \end{array}\right] \end{array}\right] \end{array}\right]$$

It corresponding to the fully-specified semantic representation of H, after eliminating @-term in the formula (5.6).

This final step corresponds to the following natural language inference.

(5.14) MONOTONICITY INFERENCE

	Most farmers who own a donkey are farmers who love a donkey.
P_2	Every farmer who loves a donkey are kind.
H	Most farmer who own a donkey are kind.

In this way, we can account for a natural language inference involving the interaction of quantifiers and anaphora.

One drawback of our analysis is that we have assumed the non-standard elimination rule of subset type that has not yet been fully examined. We could have avoided using this rule if we had used injection instead of bijection in the definition. It is straightforward to prove that Sundholm's original definition of *most* is right upward monotone. However, as we have mentioned earlier, it becomes difficult to generalize the definition to downward entailing quantifiers. The exact cardinal number can also be used for the anaphora resolution of the subsequent discourse, especially in complement anaphora.

One may think that another possibility is to get rid of the bijection relativized to **entity** and investigate the other solutions to the proportion problem. For instance, as Ranta (1994) mentioned, we can use subset types in the semantic representation of *most*. The subset type counterpart of the semantic representation is given as follows.

$$(5.15)\quad \left[\begin{array}{l} k : \mathbf{Nat} \\ \left[\begin{array}{l} k \geq \mathbf{majorityOf}\left(\pi_1\pi_1\left(@ :: \left[\begin{array}{l} a : \mathbf{Finite}_{\pi_1}(\{x : \mathbf{entity} \mid A\}) \\ \pi_1(a) \geq 1 \end{array}\right]\right)\right) \\ \left[\begin{array}{l} f : \mathbf{M}_k \rightarrow \left\{x : \mathbf{entity} \mid \left[\begin{array}{l} u : A \\ B \end{array}\right]\right\} \\ \mathbf{bijection}\,(f) \end{array}\right] \end{array}\right] \end{array}\right]$$

The alternative is to introduce the notion of a *mere proposition* as is proposed by Luo (2020) and represent the indefinite in the restrictor by the weak version of existential quantifier, $\exists$. With a notion of mere proposition, one can adopt the principle of *proof irrelevance* stating that any two proofs of the same logical proposition should be the same. Using the logical operator $\exists$ can thus prevent the multiple-donkey problem. According to their proposal, the donkey sentence in existential reading is represented as follows.

$$(5.16)\quad \mathbf{Most}\, z : [\Sigma x : \mathbf{Farmer}\, \exists y : \mathbf{Donkey}.\, \mathbf{own}(x, y)].$$
$$\exists y' : [\Sigma y : \mathbf{Donkey}.\, \mathbf{own}(\pi_1 z, y)].\, \mathbf{beat}(\pi_1 z, \pi_1 y')$$

Here, $\exists$ in the first line ensures the correct counting, while Σ in the second line provides an antecedent to the donkey pronoun.

Analyses in this direction can avoid using Theorem 3, as $\mathbf{bijection}_{\pi_1}$ becomes unnecessary. They cannot, however, immediately get rid of Theorem 5. We used the theorem to show $(x : \mathbf{entity}) \times (u : A) \times B'$'s finiteness from $(x : \mathbf{entity}) \times A$, not from $(x : \mathbf{entity}) \times (u : A) \times B$. The reason we preferred this direction is related to the difficulties of constructing bijection,

which requires one to care about the newly involved elements.

Moreover, while these analyses can ignore the indefinite of restrictor when counting, they also block an anaphoric link from subsequent sentences, which is crucially needed in plural anaphora. In (5.15), one can obtain $f(z)$ of type $\{x : \mathbf{entity} \mid (u : A) \times B\}$ by applying some $z : \mathbf{M}_k$ to f, but an entity appearing in the specification is not accessible. In (5.16), the donkey introduced by a Σ-type is only accessible from the scope of the variable y', which is merely logical operator.

Here, the tension between the two aspects of quantificational expressions, i.e., logical and linguistic characteristics, is highlighted again. If we focus on the role of "counting" in quantifier, indefinite information becomes something that wants to be hidden. On the other hand, this information is essential for the anaphora of natural language, which requires a counting method that can maintain it.

Chapter 6

Concluding Summary

This thesis aimed to provide an analysis of quantifiers in natural language in the framework of DTS. We have tried to account for the interaction of quantifier meaning and dynamic linguistic phenomena, which has been discussed in model-theoretic semantics, from the viewpoint of proof-theoretic semantics based on dependent type theory.

We adopted the explicit approach based on Sundholm (1989)'s previous study to give a semantic representation of the quantifier *most*. The operation of counting, which plays a central role in quantifiers' meaning, was defined as a function. By representing the quantified objects using a Σ-type, the semantic representation retained the quantifier's internally dynamic nature. On the other hand, to avoid the proportion problem caused by pair quantification, we adopt injections and bijections different from the standard one as Sundholm's proposal. One of the major updates from the previous proposal is that the quantifier's meaning is defined by bijection. It enables maximizing the cardinal number and introducing the desired antecedent for the plural pronoun. Furthermore, it allows generalization to other quantifiers such as *at most n* and *exactly*

n. We also proposed a semantic representation that accounts for the ambiguity between existential and universal readings in donkey sentences. The two readings are derived from the ambiguity of proof construction for the donkey pronoun.

Furthermore, these quantifiers' semantic representation became the source of the dependent interpretation observed in plural anaphora and quantificational subordination. The plural pronoun itself was defined simply as a pronoun whose antecedent is a plural object, i.e., more than one entity. Then, the function introduced by the quantifier can be the antecedent of the plural pronoun by the proof search procedure of DTS. The function introduced by the quantifier represents a dependency relation between antecedent objects in terms of the structure of dependent types. This function contributes to the anaphora resolution of the singular pronoun, which can account for its dependent interpretation.

In Chapter 5, we showed that the quantifiers we defined satisfy the inferential properties. From a proof-theoretic perspective, we formulated *conservativity* and *right monotonicity* as theorems in dependent type theory. As the theorems also consider anaphoric dependency, they can account for natural language inference involving anaphora.

The property of right monotonicity was proved in two steps. In the first step, we showed the existence of cardinal numbers. In the second step, we proved the right upward monotonicity for quantifiers whose cardinal number is defined as greater than or equal to some natural number $\mathcal{N}$. Similarly, the right downward monotone property is proved for quantifiers whose cardinal number is less than or equal to $\mathcal{N}$. For the left monotonicity, the same definition as conservativity and right monotonicity is not sufficient to account for anaphoric dependency in the quantified

sentence. Instead of defining the left argument's inclusion relation as a logical entailment, a possible direction is to consider the type's nature in dependent type theory. The full analysis is left for future work.

One contribution of this study is that it extended the empirical coverage of DTS by analyzing quantifiers in the framework. However, it also shows that this framework provides a basis for a unified analysis of quantifiers' basic meaning, interaction with dynamic linguistic phenomena, and inferential properties, suggesting that DTS offers a powerful mechanism for analyzing the natural language meaning. Furthermore, this study provided new evidence that DTS, and possibly other proof-theoretic semantics based on dependent type theory, can give an alternative view on various linguistic phenomena discussed in model-theoretic semantics.

One of the future directions is to make the semantic representation of the quantifier given here even richer. For example, the meaning of the quantifier *most* adopted here was a classical one, defined by the majority of restrictor cardinal numbers. This static definition could be improved to a context-dependent definition or a proportion-based definition. It is also possible to give an analysis that takes domain restriction into account. Furthermore, the analysis of quantifiers other than type $\langle 1, 1 \rangle$ and the extension of the analysis to plural are future tasks.

One of the future works concerning DTS is a detailed study on the nature of proof search. DTS's proof search is a powerful mechanism that contributes significantly to the analysis of intra-sentential and discourse anaphora. It has also been observed that there are cases where inference is necessary for anaphora resolution and presupposition binding. On the other hand, to prevent excessive inferences and account for unaccessible cases, we need a more systematic analysis of how the proof search can

be restricted. Furthermore, investigating the relationship of inference in anaphora resolution procedure and inference in entailment and general inference is left for future work.

Appendices

Appendix A: Formal system.

Definition 16 (Alphabet for $\lambda_{\Pi\Sigma}$)**.** An alphabet for $\lambda_{\Pi\Sigma}$ is a $\langle Var, Con\rangle$, where Var is a set of variables, and Con is a set of constants. Dependent type semantics employs an alphabet as follows:

$$Var \overset{def}{\equiv} \{x, y, z, u, v, ...\}$$
$$Con \overset{def}{\equiv} \{\textbf{entity}, \textbf{book}, \textbf{arrive}, ... john, mary, ...\}$$

Definition 17 (Preterms)**.** The collection of preterms is recursively defined as follows (where $x \in Var$ and $c \in Con$, $i = 0, 1, 2, ...$).

$$\begin{aligned}\Lambda := {} & x \mid c \mid @_i \mid \textbf{type} \mid \mathsf{kind} \mid \Lambda :: \Lambda \mid (x : \Lambda) \to \Lambda \mid \lambda x.\Lambda \mid \Lambda\Lambda \\ & \mid (x : \Lambda) \times \Lambda \mid \langle \Lambda, \Lambda\rangle \mid \pi_1(\Lambda) \mid \pi_2(\Lambda) \mid \{\Lambda, ..., \Lambda\} \mid \mathsf{case}_\Lambda(\Lambda, ..., \Lambda) \\ & \mid \bot \mid \top \mid \langle\rangle \mid \Lambda \uplus \Lambda \mid \mathsf{inl}(\Lambda) \mid \mathsf{inr}(\Lambda) \mid \mathsf{case}\ \Lambda\ \mathsf{of}\ (\Lambda, \Lambda) \\ & \mid \Lambda =_\Lambda \Lambda \mid \mathsf{refl}_\Lambda(\Lambda) \mid \mathsf{idpeel}(\Lambda, \Lambda) \mid \textbf{Nat} \mid 0 \mid \mathsf{suc}(\Lambda) \mid \mathsf{natrec}(\Lambda, \Lambda, \Lambda)\end{aligned}$$

Definition 18 (Signature)**.** A signature σ is defined recursively as follows.

$$\sigma := () \mid \sigma, c : A$$

where $()$ is an empty signature, $c \in Con$, $A \in \Lambda$ s.t. $\vdash_\sigma A : \textbf{type}$.

Definition 19 (Context)**.** A context is defined recursively as follows.

$$\Gamma := \ () \mid \Gamma,\ x : A$$

where () is an empty context, $x \in Var$, $A \in \Lambda$ s.t. $\Gamma \vdash_\sigma A :$ **type**.

Definition 20 (Constant symbol rule)**.** For any $(c : A) \in \sigma$,

$$\frac{}{c : A}\ con$$

Definition 21 (Type rules)**.**

$$\frac{}{\textbf{type} : \mathsf{kind}}\ \textbf{type}F$$

Definition 22 (Π-type)**.** For any $i \in \mathbb{N}$,
$(\mathsf{s}_1, \mathsf{s}_2) \in \{(\textbf{type}, \textbf{type}),\ (\textbf{type}, \mathsf{kind}),\ (\mathsf{kind}, \textbf{type}),\ (\mathsf{kind}, \mathsf{kind})\}$,
and $\mathsf{s} \in \{\textbf{type}, \mathsf{kind}\}$

$$\frac{\begin{array}{c}\overline{x : A}^{\,\mathrm{i}} \\ \vdots \\ A : \mathsf{s}_1 \quad B : \mathsf{s}_2\end{array}}{(x : A) \to B : \mathsf{s}_2}\ \Pi F, \mathrm{i} \qquad \frac{\begin{array}{c}\overline{x : A}^{\,\mathrm{i}} \\ \vdots \\ A : \mathsf{s} \quad M : B\end{array}}{\lambda x.M : (x : A) \to B}\ \Pi I, \mathrm{i}$$

$$\frac{M : (x : A) \to B \quad N : A}{MN : B[N/x]}\ \Pi E$$

$$\frac{A \to_\beta A'}{(x : A) \to B \to_\beta (x : A') \to B} \qquad \frac{B \to_\beta B'}{(x : A) \to B \to_\beta (x : A) \to B'}$$

$$\frac{M \to_\beta M'}{\lambda x.M \to_\beta \lambda x.M'}$$

$$\frac{M \to_\beta M'}{MN \to_\beta M'N} \qquad \frac{N \to_\beta N'}{MN \to_\beta MN'}$$

$$(\lambda x.M)N \to_\beta M[N/x]$$

Definition 23 (Implication)**.**

$$A \to B \overset{def}{\equiv} (x : A) \to B \quad \text{where } x \notin fv(B)$$

Derived rule 24. For any $i \in \mathbb{N}$,

$$\frac{\begin{array}{c}\overline{x : A}^{\,\mathrm{i}} \\ \vdots \\ A : \mathsf{s} \quad M : B\end{array}}{\lambda x.M : A \to B}\ \to I, \mathrm{i} \quad \text{where } x \notin fv(B) \qquad \frac{M : A \to B \quad N : A}{MN : B}\ \to E$$

Definition 25 (Σ-type)**.** For any $i \in \mathbb{N}$
and $(\mathsf{s}'_1, \mathsf{s}'_2) \in \{(\mathbf{type}, \mathbf{type}),\ (\mathbf{type}, \mathsf{kind}),\ (\mathsf{kind}, \mathsf{kind})\}$,

$$\frac{\begin{array}{c}\overline{x : A}^{\,\mathrm{i}} \\ \vdots \\ A : \mathsf{s}'_1 \quad B : \mathsf{s}'_2\end{array}}{(x : A) \times B : \mathsf{s}'_2}\ \Sigma F, \mathrm{i} \qquad \frac{M : A \quad N : B[M/x]}{\langle M, N \rangle : (x : A) \times B}\ \Sigma I$$

$$\frac{M : (x : A) \times B}{\pi_1(M) : A}\ \Sigma E_L \qquad \frac{M : (x : A) \times B}{\pi_2(M) : B[\pi_1(M)/x]}\ \Sigma E_R$$

$$\frac{A \to_\beta A'}{(x : A) \times B \to_\beta (x : A') \times B} \qquad \frac{B \to_\beta B'}{(x : A) \times B \to_\beta (x : A) \times B'}$$

$$\frac{M \to_\beta M'}{\langle M, N \rangle \to_\beta \langle M', N \rangle} \qquad \frac{N \to_\beta N'}{\langle M, N \rangle \to_\beta \langle M, N' \rangle}$$

$$\frac{M \to_\beta M'}{\pi_1 M \to_\beta \pi_1 M'}$$

$$\pi_i(\langle M_1, M_2 \rangle) \to_\beta M_i \quad \text{where } i \in \{1, 2\}$$

Definition 26 (Conjunction)**.**

$$(A \times B) \stackrel{def}{\equiv} (x : A) \times B \quad \text{where } x \notin fv(B)$$

Derived rule 27.

$$\frac{M : A \quad N : B}{\langle M, N \rangle : (A \times B)}\ \times I \qquad \frac{M : (A \times B)}{\pi_1(M) : A}\ \times E_L \qquad \frac{M : (A \times B)}{\pi_2(M) : B}\ \times E_R$$

Definition 28 (Enumeration type)**.** For any $n \in \mathbb{N}$, each i such that $1 \leq i \leq n$, and $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$

$$\frac{}{\{a_1, ..., a_n\} : \mathbf{type}}\ \{\}F \qquad \frac{}{a_i : \{a_1, ..., a_n\}}\ \{\}I$$

$$\frac{M : \{a_1, ..., a_n\} \quad C : \{a_1, ..., a_n\} \to \mathsf{s} \quad N_1 : C(a_1) \quad ... \quad N_n : C(a_n)}{\mathsf{case}_M(N_1, ..., N_n) : C(M)} \{\}E$$

$$\mathsf{case}_{a_i}(N_1, ..., N_i, ..., N_n) \to_\beta N_i \quad \text{where } a_i : \{a_1, ..., a_i, ...a_n\}$$

Bottom type and Top type is defined as follows:

$$\bot \stackrel{def}{\equiv} \{\}$$

$$\top \stackrel{def}{\equiv} \{\langle\rangle\}$$

Definition 29 (Bottom type)**.** For $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$,

$$\frac{}{\bot : \mathbf{type}} \bot F \qquad \frac{M : \bot \quad C : \bot \to \mathsf{s}}{\mathsf{case}_M() : C(M)} \bot E$$

Definition 30 (Negation)**.**

$$\neg A \stackrel{def}{\equiv} A \to \bot$$

Definition 31 (Top type)**.** For $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$,

$$\frac{}{\top : \mathbf{type}} \top F \qquad \frac{}{\langle\rangle : \top} \top I$$

$$\frac{M : \top \quad C : \top \to \mathsf{s} \quad N : C(\langle\rangle)}{\mathsf{case}_M(N) : C(M)} \top E$$

Definition 32 (Disjoint union of two types)**.** For any $i \in \mathbb{N}$ and $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$,

$$\frac{A : \mathsf{s} \quad B : \mathsf{s}}{A \uplus B : \mathsf{s}} \uplus F$$

$$\frac{M : A \quad B : \mathsf{s}}{\mathsf{inl}(M) : A \uplus B} \uplus I_L \qquad \frac{A : \mathsf{s} \quad N : B}{\mathsf{inr}(N) : A \uplus B} \uplus I_R$$

$$\frac{L : A \uplus B \quad C : A \uplus B \to \mathsf{s} \quad \begin{array}{c}\overline{\begin{array}{c}x : A\\ \vdots\\ M : C(\mathsf{inl}(x))\end{array}}\,\mathrm{i} \quad \overline{\begin{array}{c}x : B\\ \vdots\\ N : C(\mathsf{inr}(x))\end{array}}\,\mathrm{i}\end{array}}{\mathsf{case}\ L\ \mathsf{of}\ (\lambda x.M, \lambda x.N) : C(L)} \uplus E, \mathrm{i}$$

$$\mathsf{case}\ \mathsf{inl}(a)\ \mathsf{of}\ (\lambda x.M, \lambda x.N) \twoheadrightarrow_\beta M(a)$$

$$\mathsf{case}\ \mathsf{inr}(b)\ \mathsf{of}\ (\lambda x.M, \lambda x.N) \twoheadrightarrow_\beta N(b)$$

Definition 33 (Intentional equality)**.** For $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$,

$$\frac{A : \mathsf{s} \quad M : A \quad N : A}{M =_A N : \mathsf{s}} =F \qquad \frac{A : \mathsf{s} \quad M : A}{\mathsf{refl}_A(M) : M =_A M} =I$$

$$\frac{E : M_1 =_A M_2 \quad C : (x : A) \to (y : A) \to (x =_A y) \to \mathsf{s} \quad R : (x : A) \to Cxx(\mathsf{refl}_A(x))}{\mathsf{idpeel}(E, R) : CM_1M_2E} =E$$

$$\mathsf{idpeel}(\mathsf{refl}_A(M), R) \to_\beta R(M)$$

Definition 34 (Natural numbers)**.** For $\mathsf{s} \in \{\mathbf{type}, \mathsf{kind}\}$,

$$\frac{}{\mathbf{Nat} : \mathbf{type}}\ \mathbf{Nat}F \qquad \frac{}{0 : \mathbf{Nat}}\ \mathbf{Nat}I_0 \qquad \frac{n : \mathbf{Nat}}{\mathsf{suc}(n) : \mathbf{Nat}}\ \mathbf{Nat}I_s$$

$$\frac{n : \mathbf{Nat} \quad C : \mathbf{Nat} \to \mathsf{s} \quad e : C(0) \quad f : (k : \mathbf{Nat}) \to C(k) \to C(\mathsf{suc}(k))}{\mathsf{natrec}(n, e, f) : C(n)}\ \mathbf{Nat}E$$

$$\begin{aligned} \mathsf{natrec}(0, e, f) &\to_\beta e \\ \mathsf{natrec}(\mathsf{suc}(n), e, f) &\to_\beta f(n)(\mathsf{natrec}(n, e, f)) \end{aligned}$$

Definition 35 (Addition)**.**

$$m + n \overset{def}{\equiv} \mathsf{natrec}(m, n, \lambda x \lambda y.\mathsf{suc}(y))$$

$$\frac{m : \mathbf{Nat} \quad n : \mathbf{Nat}}{m + n : \mathbf{Nat}}\ +F$$

Definition 36 ($\geq$)**.**

$$m \geq n \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ m =_{\mathbf{Nat}} n + k \end{array} \right]$$

$$n \leq m \overset{def}{\equiv} \left[\begin{array}{l} k : \mathbf{Nat} \\ m =_{\mathbf{Nat}} n + k \end{array} \right]$$

Definition 37 (Multiplication)**.**

$$m \cdot n \overset{def}{\equiv} \mathsf{natrec}(m, 0, \lambda x \lambda y.(y + n))$$

$$\frac{m : \mathbf{Nat} \quad n : \mathbf{Nat}}{m \cdot n : \mathbf{Nat}}\ \cdot F$$

Definition 38 (@-rule)**.** For any $i \in \mathbb{N}$,

$$\frac{A : \mathbf{type} \quad A \ \ true}{(@_i :: A) : A}\ @$$

Definition 39 (α-conversion)**.** For any $x, y \in Var$ and $A, B, M \in \Lambda$,

$$\begin{aligned}(x : A) \to B &\equiv (y : A) \to B[y/x] \quad \text{where } y \notin fv(B)\\ (x : A) \times B &\equiv (y : A) \times B[y/x] \quad \text{where } y \notin fv(B)\\ \lambda x.M &\equiv \lambda y.M[y/x] \quad \text{where } y \notin fv(M)\end{aligned}$$

Definition 40 (More-than-zero-steps reduction $\twoheadrightarrow_\beta$)**.**

$$\frac{}{M \twoheadrightarrow_\beta M} \qquad \frac{L \to_\beta M \quad M \twoheadrightarrow_\beta N}{L \twoheadrightarrow_\beta N}$$

Derived rule 41.

$$\frac{L \twoheadrightarrow_\beta M \quad M \twoheadrightarrow_\beta N}{L \twoheadrightarrow_\beta N}$$

Definition 42 (Subject Reduction)**.**

$$\Gamma \vdash M : A,\ M \to_\beta M' \implies \Gamma \vdash M' : A$$

Definition 43 ($\equiv_\beta$)**.**

$$\frac{}{M \equiv_\beta M} \qquad \frac{L \to_\beta M \quad M \equiv_\beta N}{L \equiv_\beta N} \qquad \frac{M \to_\beta L \quad M \equiv_\beta N}{L \equiv_\beta N}$$

Derived rule 44.

$$\frac{L \twoheadrightarrow_\beta M \quad M \equiv_\beta N}{L \equiv_\beta N} \qquad \frac{M \twoheadrightarrow_\beta L \quad M \equiv_\beta N}{L \equiv_\beta N}$$

Derived rule 45.

$$\frac{M \equiv_\beta N}{N \equiv_\beta M} \qquad \frac{L \equiv_\beta M \quad M \equiv_\beta N}{L \equiv_\beta N}$$

Definition 46 (Conversion Rule)**.**

$$\frac{M : A \quad A \equiv_\beta B}{M : B}\ conv$$

Definition 47 (Weakening)**.**

$$\frac{M : A \quad N : B}{M : A}\ wk$$

Definition 48 (Notation)**.**

$$(x : A) \times B \overset{def}{\equiv} \left[\begin{array}{l} x : A \\ B \end{array}\right]$$

$$(A \times B) \overset{def}{\equiv} \left[\begin{array}{l} A \\ B \end{array}\right]$$

$$(x_0, x_1, ..., x_n : A) \to B \overset{def}{\equiv} (x_0 : A) \to (x_1 : A) \to ... \to (x_n : A) \to B$$

$$id_A \overset{def}{\equiv} \lambda x.x \quad \text{where } x : A$$

Appendix B: Theorems.

Theorem 12

$$\frac{A : \mathbf{type} \quad B : \mathbf{type} \quad C : \mathbf{type} \quad M : A \uplus B}{\mathbf{p}_{12}(M) : (A \uplus C) \uplus B}$$

Theorem 13: Symmetry law for $=$

$$\frac{A : \mathsf{s} \quad Q : M =_A N}{\mathbf{p}_{13}(Q) : N =_A M} =\text{Sym}$$

Theorem 14: Transitivity law for $=$

$$\frac{A : \mathsf{s} \quad L : A \quad M : A \quad N : A \quad Q : L =_A M \quad R : M =_A N}{\mathbf{p}_{14}(Q;R) : L =_A N} =\text{Trans}$$

Theorem 15: Substitution by $=$

$$\frac{A : \mathsf{s} \quad M : A \quad N : A \quad P : A \to \mathsf{s} \quad Q : M =_A N \quad R : PM}{\mathbf{p}_{15}(Q;R) : PN} =\text{Sub}$$

Theorem 16

$$\frac{A : \mathsf{s} \quad M : A \quad N : A \quad M \equiv_\beta N}{\mathsf{refl}_A(M) : M =_A N}$$

$$A : \mathsf{s},\ M : A,\ N : A, \vdash M =_A N\ \mathit{true} \implies M \equiv_\beta N$$

Theorem 17

$$\begin{aligned} 0 + n &\equiv_\beta n \\ \mathsf{suc}(m) + n &\equiv_\beta \mathsf{suc}(m + n) \end{aligned}$$

Theorem 18

$$\frac{m : \mathbf{Nat} \quad n : \mathbf{Nat}}{\mathbf{p}_{18}(m;n) : m + n =_{\mathbf{Nat}} n + m}$$

Theorem 19

$$\frac{n : \mathbf{Nat} \quad m : \mathbf{Nat} \quad M : \mathsf{suc}(n) \leq m}{\mathbf{p}_{19}(M) : n \leq m}$$

Proof. We prove that we have $\mathbf{p}_{19}(M)$.

$$\begin{aligned}
\mathbf{p}_{19}(M) &\overset{def}{\equiv} \langle \mathsf{suc}(\pi_1(M)), \mathbf{p}_{14}(\pi_2(M); \mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n)))\rangle \\
\mathbf{q} &\overset{def}{\equiv} \mathbf{p}_{14}(\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n) + \pi_1(M)); \mathbf{r}) \\
\mathbf{r} &\overset{def}{\equiv} \mathbf{p}_{15}(\mathbf{p}_{18}(n; \pi_1(M)); \mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n + \pi_1(M))))
\end{aligned}$$

$$\dfrac{\dfrac{M : \left[\begin{array}{l} k : \mathbf{Nat} \\ m =_{\mathbf{Nat}} \mathsf{suc}(n) + k \end{array}\right]}{\dfrac{\pi_1(M) : \mathbf{Nat}}{\mathsf{suc}(\pi_1(M)) : \mathbf{Nat}}\,\mathbf{Nat}I_s}\,\Sigma E_L \qquad \begin{array}{c}\vdots\ \mathcal{D}_1 \\ \mathbf{p}_{14}(\pi_2(M); \mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n))) : \\ m =_{\mathbf{Nat}} n + \mathsf{suc}(\pi_1(M))\end{array}}{\langle \mathsf{suc}(\pi_1(M)), \mathbf{p}_{14}(\pi_2(M); \mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n)))\rangle : \left[\begin{array}{l} n' : \mathbf{Nat} \\ m =_{\mathbf{Nat}} n + n' \end{array}\right]}\,\Sigma I$$

$\mathcal{D}_1$:

$$\dfrac{\overline{\mathbf{Nat} : \mathbf{type}}\,\mathbf{Nat}F \quad m : \mathbf{Nat} \quad \mathsf{suc}(n) + \pi_1(M) : \mathbf{Nat} \quad n + \mathsf{suc}(\pi_1(M)) : \mathbf{Nat} \quad \dfrac{M : \left[\begin{array}{l} k : \mathbf{Nat} \\ m =_{\mathbf{Nat}} \mathsf{suc}(n) + k \end{array}\right]}{\pi_2(M) : m =_{\mathbf{Nat}} \mathsf{suc}(n) + \pi_1(M)}\,\Sigma E_R \quad \begin{array}{c}\vdots\ \mathcal{D}_2 \\ \mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n)) : \\ \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} n + \mathsf{suc}(\pi_1(M))\end{array}}{\mathbf{p}_{14}(\pi_2(M); \mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n))) : m =_{\mathbf{Nat}} n + \mathsf{suc}(\pi_1(M))}\,=\mathrm{Trans}$$

$\mathcal{D}_2$:

$$\dfrac{\overline{\mathbf{Nat} : \mathbf{type}}\,\mathbf{Nat}F \quad \mathsf{suc}(n) + \pi_1(M) : \mathbf{Nat} \quad \mathsf{suc}(\pi_1(M)) + n : \mathbf{Nat} \quad n + \mathsf{suc}(\pi_1(M)) : \mathbf{Nat} \quad \mathbf{q} : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M)) + n \quad \begin{array}{c}\vdots\ \mathcal{D}_3 \\ \mathsf{suc}(\pi_1(M)) : \mathbf{Nat} \quad n : \mathbf{Nat}\end{array}\,(\mathrm{Th}18) \quad \begin{array}{c}\mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n) : \\ \mathsf{suc}(\pi_1(M)) + n =_{\mathbf{Nat}} n + \mathsf{suc}(\pi_1(M))\end{array}}{\mathbf{p}_{14}(\mathbf{q}; \mathbf{p}_{18}(\mathsf{suc}(\pi_1(M)); n)) : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} n + \mathsf{suc}(\pi_1(M))}\,=\mathrm{Trans}$$

$\mathcal{D}_3$:

$$\dfrac{\dfrac{\overline{\mathbf{Nat} : \mathbf{type}}\,\mathbf{Nat}F \quad \mathsf{suc}(n) + \pi_1(M) : \mathbf{Nat} \quad \mathsf{suc}(n) + \pi_1(M) : \mathbf{Nat} \quad \mathsf{suc}(\pi_1(M)) + n}{\dfrac{\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n) + \pi_1(M)) : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(n) + \pi_1(M)}{\mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(n) + \pi_1(M)}\,=I \quad \dfrac{\mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(n) + \pi_1(M)}{\equiv_{\beta} \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(n + \pi_1(M))}\,conv}}{\dfrac{\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n) + \pi_1(M)) : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(n + \pi_1(M)) \quad \mathbf{r} : \mathsf{suc}(n + \pi_1(M)) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M) + n)}{\mathbf{p}_{14}(\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n) + \pi_1(M)); \mathbf{r}) : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M) + n)}\,=\mathrm{Trans} \quad \begin{array}{c}\mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M) + n) \\ \equiv_{\beta} \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M)) + n\end{array}\,conv}}{\mathbf{p}_{14}(\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n) + \pi_1(M)); \mathbf{r}) : \mathsf{suc}(n) + \pi_1(M) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M)) + n}\,conv$$

$\mathcal{D}_4$:

$$\dfrac{\overline{\mathbf{Nat} : \mathbf{type}}\,\mathbf{Nat}F \quad n + \pi_1(M) : \mathbf{Nat} \quad \pi_1(M) + n : \mathbf{Nat} \quad \begin{array}{c}\vdots\ \mathcal{T}_1 \\ \lambda x.\mathsf{suc}(n + \pi_1(M)) =_{\mathbf{Nat}} \mathsf{suc}(x) : \\ \mathbf{Nat} \to \mathbf{type}\end{array} \quad \dfrac{n : \mathbf{Nat} \quad \pi_1(M) : \mathbf{Nat}}{\begin{array}{l}\mathbf{p}_{18}(n; \pi_1(M)) : \\ n + \pi_1(M) =_{\mathbf{Nat}} \pi_1(M) + n\end{array}}\,(\mathrm{Th}18) \quad \dfrac{\mathbf{Nat} : \mathbf{type} \quad \mathsf{suc}(n + \pi_1(M)) : \mathbf{Nat}}{\begin{array}{l}\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n + \pi_1(M))) : \\ \mathsf{suc}(n + \pi_1(M)) =_{\mathbf{Nat}} \mathsf{suc}(n + \pi_1(M))\end{array}}\,=I}{\mathbf{p}_{15}(\mathbf{p}_{18}(n; \pi_1(M)); \mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(n + \pi_1(M)))) : \mathsf{suc}(n + \pi_1(M)) =_{\mathbf{Nat}} \mathsf{suc}(\pi_1(M) + n)}\,=\mathrm{Sub}$$

$\mathcal{T}_1$:

$$\vdots$$
$$\lambda x.\mathsf{suc}(n + \pi_1(M)) =_{\mathbf{Nat}} \mathsf{suc}(x) : \mathbf{Nat} \to \mathbf{type}$$

□

Lemma 20: $\lambda z.\mathsf{inl}(z)$ is an injection, that is, the following holds.

$$\big(A, B : \mathbf{type}\big) \to (y, y' : A) \to \mathsf{inl}(y) =_{A \uplus B} \mathsf{inl}(y') \to y =_A y'$$

Proof. We prove $y =_A y'$ by assuming $A, B : \mathbf{type}$, $y, y' : A$, and $u : \mathsf{inl}(y) =_{A \uplus B} \mathsf{inl}(y')$. The following derivation holds.

$$\dfrac{\dfrac{y : A \quad B : \mathbf{type}}{\mathsf{inl}(y) : A \uplus B}\uplus I \qquad A : \mathbf{type} \qquad \dfrac{\dfrac{\overline{a : A}^{\,1} \quad \overline{a : A}^{\,1}}{a : A}\,wk \qquad \dfrac{\overline{b : B}^{\,1} \quad y : A}{y : A}\,wk}{\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y) : A}\uplus E, 1$$

Here,

$$\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y) \equiv_\beta y$$

holds, from which we have

$$\mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y)) : \mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y) =_A y \tag{6.1}$$

by Theorem 16. By applying =Sub with the assumptions $u : \mathsf{inl}(y) = \mathsf{inl}(y')$,

$$\mathbf{p}_{15}(u, \mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y))) : \mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y) =_A y.$$

By =Sym,

$$\mathbf{p}_{13}(\mathbf{p}_{15}(u, \mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y))))$$
$$: y =_A \mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y). \tag{6.2}$$

On the other hand, we also have

$$\dfrac{\dfrac{y' : A \quad B : \mathbf{type}}{\mathsf{inl}(y') : A \uplus B}\uplus I \qquad A : \mathbf{type} \qquad \dfrac{\dfrac{\overline{a : A}^{\,1} \quad \overline{a : A}^{\,1}}{a : A}\,wk \qquad \dfrac{\overline{b : B}^{\,1} \quad y : A}{y : A}\,wk}{\mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y) : A}\uplus E, 1}{},$$

which yields to

$$\mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y)) : \mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y) =_A y' \tag{6.3}$$

in the same way as the above. Thus, by applying =Trans with (6.2) and (6.3), we get

$$\begin{gathered}\mathbf{p}_{14}(\mathbf{p}_{13}(\mathbf{p}_{15}(u, \mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y)\ \mathsf{of}\ (\lambda a.a, \lambda b.y)))),\\ \mathsf{refl}_A(\mathsf{case}\ \mathsf{inl}(y')\ \mathsf{of}\ (\lambda a.a, \lambda b.y)))\\ : y =_A y'\end{gathered}$$

□

Lemma 21: $\lambda z.\mathsf{inr}(z)$ is an injection, that is, the following holds.

$$(A, B : \mathbf{type}) \to (y, y' : B) \to \mathsf{inr}(y) =_{A \uplus B} \mathsf{inr}(y') \to y =_B y'$$

Proof. The proof is given in the same manner as Lemma 20. □

Lemma 22: For any injection f_1, f_2, the composition $f_2 \circ f_1$ is an injection, that is, the following holds.

$$(A, B, C : \mathbf{type}) \to \left(u_1 : \begin{bmatrix} f_1 : A \to B \\ \mathbf{injection}\,(f_1) \end{bmatrix}\right) \to \left(u_2 : \begin{bmatrix} f_2 : B \to C \\ \mathbf{injection}\,(f_2) \end{bmatrix}\right) \to \mathbf{injection}\,(\pi_1(u_2) \circ \pi_1(u_1))$$

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{u_1 : \begin{bmatrix} f_1 : A \to B \\ \mathbf{injection}\,(f_1) \end{bmatrix}}{\pi_1 u_1 : A \to B}\Sigma E_L \quad y : A}{\pi_1 u_1(y) : B}\Pi E \quad \dfrac{\begin{matrix} u_1 : \dots \quad y' : A \\ \vdots \\ \pi_1 u_1(y') : B \end{matrix} \quad \dfrac{u_2 : \begin{bmatrix} f_2 : B \to C \\ \mathbf{injection}\,(f_2) \end{bmatrix}}{\pi_2 u_2 : (y : B) \to (y' : B) \to \pi_1 u_2(y) =_C \pi_1 u_2(y') \to y =_B y'}\Sigma E_R}{\Pi E^*}}{(\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')) : (\pi_1 u_2)((\pi_1 u_1)(y)) =_C (\pi_1 u_2)((\pi_1 u_1)(y')) \to (\pi_1 u_1)(y) =_B (\pi_1 u_1)(y')} \quad \dfrac{\begin{matrix} u_3 : (\pi_1 u_2 \circ \pi_1 u_1)(y) \\ = (\pi_1 u_2 \circ \pi_1 u_1)(y') \end{matrix} \quad \dfrac{\overline{\begin{matrix} t_1 : (\pi_1 u_2 \circ \pi_1 u_1)(y) \\ = (\pi_1 u_2)((\pi_1 u_1)(y)) \end{matrix}} \quad \overline{\begin{matrix} t_2 : (\pi_1 u_2 \circ \pi_1 u_1)(y') \\ = (\pi_1 u_2)((\pi_1 u_1)(y')) \end{matrix}}}{u_3 : (\pi_1 u_2)((\pi_1 u_1)(y)) = (\pi_1 u_2)((\pi_1 u_1)(y'))}\,conv^*}{(\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y'))(u_3) : (\pi_1 u_1)(y) =_B (\pi_1 u_1)(y')}\to E$$

Thus,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\overline{y : A}^{\,2}}\quad\dfrac{}{\overline{y' : A}^{\,2}}\quad\dfrac{\overline{u_1 : \ldots}^{\,1}}{\pi_2 u_1 : (y : A) \to (y' : A) \to \pi_1 u_1(y) =_B \pi_1 u_1(y') \to y =_A y'}\,\Sigma E_R}{(\pi_1 u_1)(y)(y') : (\pi_1 u_1)(y) =_B (\pi_1 u_1)(y') \to y =_A y'}\,\Pi E^*\quad\overline{u_1}^{\,1}\ \overline{u_2}^{\,1}\ \overline{y}^{\,2}\ \overline{y'}^{\,2}\ \overline{u_3}^{\,3}}{((\pi_1 u_1)(y)(y'))((\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')))(u_3) : y =_A y'}\,{\to}E}{\lambda u_3.((\pi_1 u_1)(y)(y'))((\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')))(u_3)) : (\pi_1(u_2) \circ \pi_1(u_1))(y) =_C (\pi_1(u_2) \circ \pi_1(u_1))(y') \to y =_A y'}\,{\to}I, 3}{\lambda y.\lambda y'.\lambda u_3.((\pi_1 u_1)(y)(y'))((\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')))(u_3)) : (y : A) \to (y' : A) \to (\pi_1(u_2) \circ \pi_1(u_1))(y) =_C (\pi_1(u_2) \circ \pi_1(u_1))(y') \to y =_A y'}\,\Pi I, 2^*}{\lambda A \lambda B \lambda C \lambda u_1 \lambda u_2.\lambda y.\lambda y'.\lambda u_3.((\pi_1 u_1)(y)(y'))((\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')))(u_3))}\,\Pi I, 1^*$$

$$: (A, B, C : \mathbf{type}) \to \left(u_1 : \left[\begin{array}{l} f_1 : A \to B \\ \mathbf{injection}\,(f_1) \end{array}\right]\right) \to \left(u_2 : \left[\begin{array}{l} f_2 : B \to C \\ \mathbf{injection}\,(f_2) \end{array}\right]\right) \to \mathbf{injection}\,(\pi_1(u_2) \circ \pi_1(u_1))$$

By the above derivation, we have

$$\lambda A \lambda B \lambda C \lambda u_1 \lambda u_2.\lambda y.\lambda y'.\lambda u_3.((\pi_1 u_1)(y)(y'))((\pi_2 u_2)(\pi_1 u_1(y))(\pi_1 u_1(y')))(u_3))$$
$$: (A, B, C : \mathbf{type}) \to \left(u_1 : \left[\begin{array}{l} f_1 : A \to B \\ \mathbf{injection}\,(f_1) \end{array}\right]\right) \to \left(u_2 : \left[\begin{array}{l} f_2 : B \to C \\ \mathbf{injection}\,(f_2) \end{array}\right]\right) \to \mathbf{injection}\,(\pi_1(u_2) \circ \pi_1(u_1))$$

□

Lemma 23 (injection $(\lambda x.\mathsf{inl}(\mathsf{inr}(x)))$**):** $\lambda x.\mathsf{inl}(\mathsf{inr}(x))$ is an injection, that is, the following holds.

$$(A, B, C : \mathbf{type}) \to (y, y' : B) \to \mathsf{inl}(\mathsf{inr}(y)) =_{(A \uplus B) \uplus C} \mathsf{inl}(\mathsf{inr}(y')) \to y =_B y'$$

Proof. It immediately follows from Lemma 20, Lemma 21, and Lemma 22.

□

Theorem 24

Assume that there exists $f : A \to B$ such that **injection** (f) and **surjection** (f) are true, where A and B are arbitrary elements of **type**. That is, the following proof term i and s exist.

$$i : \quad (y : A) \to (y' : A) \to f(y) =_B f(y') \to y =_A y'$$
$$s : \quad (z : B) \to \left[\begin{array}{l} y : A \\ z =_B f(y) \end{array}\right]$$

Then, for $\pi_1 s : B \to A$:

(1) **injection** $(\pi_1 s)$ and **surjection** $(\pi_1 s)$ are true

(2) $\lambda x.f(\pi_1 s(x)) : B \to B$ and $\lambda x.(\pi_1 s(f(x))) : A \to A$ are identity functions

Proof. (1) We prove that **injection** $(\pi_1 s)$ and **surjection** $(\pi_1 s)$ are true.

<u>**injection** $(\pi_1 s)$</u>

$$\dfrac{s:(z:B)\to\begin{bmatrix}y:A\\ z=f(y)\end{bmatrix} \quad \overline{z:B}^{\,3}}{s(z):\begin{bmatrix}y:A\\ z=_B f(y)\end{bmatrix}}\,\Pi E \qquad \dfrac{s(z):\begin{bmatrix}y:A\\ z=_B f(y)\end{bmatrix}}{\pi_2(s(z)):z=_B f(\pi_1 s(z))}\,\Sigma E_L$$

$$\dfrac{A:\mathbf{type} \quad \dfrac{}{u:\pi_1 s(z)=_A \pi_1 s(z')}\,1}{\mathbf{p}_{13}(u):\pi_1 s(z')=_A \pi_1 s(z)}\,{=}\mathrm{Sym} \qquad \dfrac{\dfrac{s:(z:B)\to\begin{bmatrix}y:A\\ z=f(y)\end{bmatrix} \quad \overline{z':B}^{\,2}}{s(z'):\begin{bmatrix}y':A\\ z'=_B f(y')\end{bmatrix}}\,\Pi E}{\pi_2(s(z')):z'=_B f(\pi_1 s(z'))}\,\Sigma E_L$$

$$\dfrac{\dfrac{A:\mathbf{type} \quad \pi_1 s(z'):A \quad \pi_1 s(z):A \quad \lambda y'.z'=_B f(y'):A\to\mathbf{type} \quad \mathbf{p}_{13}(u):\pi_1 s(z')=_A \pi_1 s(z) \quad \pi_2(s(z')):z'=_B f(\pi_1 s(z'))}{\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z'))):z'=_B f(\pi_1 s(z))}\,{=}\mathrm{Sub}}{\mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z')))):f(\pi_1 s(z))=_B z'}\,{=}\mathrm{Sym}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{B:\mathbf{type} \quad \overline{z:B}^{\,3} \quad f(\pi_1 s(z)):B \quad \overline{z':B}^{\,2} \quad \pi_2(s(z)):z=_B f(\pi_1 s(z)) \quad \mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z')))):f(\pi_1 s(z))=_B z'}{\mathbf{p}_{14}(\pi_2(s(z)),\mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z'))))):z=_B z'}\,{=}\mathrm{Trans}}{\lambda u.\mathbf{p}_{14}(\pi_2(s(z)),\mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z'))))):\pi_1 s(z)=_A \pi_1 s(z')\to z=_B z'}\,\Pi I,1}{\lambda z'.\lambda u.\mathbf{p}_{14}(\pi_2(s(z)),\mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z'))))):(z':B)\to\pi_1 s(z)=_A \pi_1 s(z')\to z=_B z'}\,\Pi I,2}{\lambda z.\lambda z'.\lambda u.\mathbf{p}_{14}(\pi_2(s(z)),\mathbf{p}_{13}(\mathbf{p}_{15}(\mathbf{p}_{13}(u),\pi_2(s(z'))))):(z:B)\to(z':B)\to\pi_1 s(z)=_A \pi_1 s(z')\to z=_B z'}\,\Pi I,3$$

<u>**surjection** $(\pi_1 s)$</u>

$$\dfrac{s:(z:B)\to\begin{bmatrix}y:A\\ z=f(y)\end{bmatrix} \quad \dfrac{f:A\to B \quad y:A}{f(y):B}\,{\to}E}{s(f(y)):\begin{bmatrix}y':A\\ f(y)=_B f(y')\end{bmatrix}}\,\Pi E \qquad \dfrac{s(f(y)):\begin{bmatrix}y':A\\ f(y)=_B f(y')\end{bmatrix}}{\pi_1(s(f(y))):A}\,\Sigma E_L \qquad \dfrac{s(f(y)):\begin{bmatrix}y':A\\ f(y)=_B f(y')\end{bmatrix}}{\pi_2(s(f(y))):f(y)=_B f(\pi_1(s(f(y))))}\,\Sigma E_R$$

$$\dfrac{\begin{array}{l}i:(y:A)\\ \quad\to(y':A)\\ \quad\quad\to f(y)=_B f(y')\\ \quad\quad\quad\to y=_A y'\end{array} \quad y:A \quad \pi_1(s(f(y))):A \quad \pi_2(s(f(y))):f(y)=_B f(\pi_1(s(f(y))))}{i(y)(\pi_1(s(f(y))))(\pi_2(s(f(y)))):y=_A \pi_1(s(f(y)))}\,\Pi E^*$$

Thus, a proof term of **surjection** $(\pi_1 s)$ can be obtained by the following delivation.

$$\dfrac{\dfrac{\dfrac{f:A\to B \quad \overline{y:A}^{\,1}}{f(y):B}\,{\to}E \qquad \begin{array}{c}i:\ldots \quad s:\ldots \quad f:\ldots \quad \overline{y:A}^{\,1}\\ \vdots\\ i(y)(\pi_1(s(f(y))))(\pi_2(s(f(y)))):y=_A \pi_1(s(f(y)))\end{array}}{\langle f(y),i(y)(\pi_1(s(f(y))))(\pi_2(s(f(y))))\rangle:\begin{bmatrix}z:B\\ y=_A \pi_1(s(z))\end{bmatrix}}\,\Sigma I}{\lambda y.\langle f(y),i(y)(\pi_1(s(f(y))))(\pi_2(s(f(y))))\rangle:(y:A)\to\begin{bmatrix}z:B\\ y=_A \pi_1(s(z))\end{bmatrix}}\,\Pi I,1$$

(2) We prove that $\lambda x.f(\pi_1 s(x)):B\to B$ and $\lambda x.(\pi_1 s(f(x))):A\to A$ are identity functions, namely, $(x:B)\to x=_B f(\pi_1 s(x))$ and $(x:A)\to x=_A \pi_1 s(f(x))$.

<u>$(x:B)\to x=_B f(\pi_1 s(x))$</u>

$$\dfrac{\dfrac{\dfrac{s : (z : B) \to \begin{bmatrix} y : A \\ z =_B f(y) \end{bmatrix} \qquad \overline{x : B}^{\,1}}{s(x) : \begin{bmatrix} y : A \\ x =_B f(y) \end{bmatrix}}\,\Pi E}{\pi_2(s(x)) : x =_B f(\pi_1 s(x))}\,\Sigma E_L}{\lambda x.\pi_2(s(x)) : (x : B) \to x =_B f(\pi_1 s(x))}$$

<u>$(x : A) \to x =_A \pi_1 s(f(x))$</u>

$$\dfrac{\dfrac{\dfrac{i : (y : A) \to (y' : A) \to f(y) =_B f(y') \to y =_A y' \quad \overline{x : A}^{\,1} \quad \dfrac{\dfrac{s : (z : B) \to \begin{bmatrix} y : A \\ z =_B f(y) \end{bmatrix} \quad \dfrac{f : A \to B \quad \overline{x : A}^{\,1}}{f(x) : B}\,\Pi E}{s(f(x)) : \begin{bmatrix} y : A \\ f(x) =_B f(y) \end{bmatrix}}\,\Pi E}{\pi_1 s(f(x)) : A}\,\Sigma E_R}{i(x)(\pi_1 s(f(x))) : f(x) =_B f(\pi_1 s(f(x))) \to x =_A \pi_1 s(f(x))}\,\Pi E^* \qquad \dfrac{\begin{array}{c}\overline{}\\ \vdots \\ s(f(x)) : \begin{bmatrix} y : A \\ f(x) =_B f(y) \end{bmatrix}\end{array}}{\pi_2 s(f(x)) : f(x) =_B f(\pi_1 s(f(x)))}\,\Sigma E_L}{i(x)(\pi_1 s(f(x)))(\pi_2 s(f(x))) : x =_A \pi_1 s(f(x))}\,\Pi E}{\lambda x.i(x)(\pi_1 s(f(x)))(\pi_2 s(f(x))) : (x : A) \to x =_A \pi_1 s(f(x))}\,\Pi I, 1$$

□

Theorem 25: Identity in Canonical form

$$\dfrac{A : \mathbf{type} \quad B : \mathbf{type} \quad z : A \quad z' : B}{\neg(\mathsf{inl}(z) =_{A \uplus B} \mathsf{inr}(z'))\ \ \mathit{true}}$$

Proof. Let **isLeft** $\overset{def}{\equiv} \lambda c.\mathsf{case}\ x\ \mathsf{of}\ (\lambda y.\overline{\top}, \lambda y.\overline{\bot})$. We have

$$\mathbf{isLeft}(\mathsf{inl}(z)) \equiv \overline{\top}$$

$$\mathbf{isLeft}(\mathsf{inr}(z')) \equiv \overline{\bot}.$$

Now we have the following derivation $((\lambda x.\lambda y.\lambda e.\mathbf{isLeft}(x) \to \mathbf{isLeft}(y))xx(\mathsf{refl}_{A \uplus B}(x)) \equiv \mathbf{isLeft}(x) \to \mathbf{isLeft}(x))$.

$$\dfrac{\begin{array}{c} z : A \quad z' : B \\ \vdots \\ \mathsf{inl}(z) = \mathsf{inr}(z') : \mathbf{type} \end{array} \quad \dfrac{\dfrac{\overline{e : \mathsf{inl}(z) = \mathsf{inr}(z')}^{\,1} \quad \dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}\overline{x : A \uplus B}\\ \vdots \\ \mathbf{isLeft}(x) : \mathbf{type}\end{array} \quad \begin{array}{c}\overline{y : A \uplus B}\\ \vdots \\ \mathbf{isLeft}(y) : \mathbf{type}\end{array}}{\mathbf{isLeft}(x) \to \mathbf{isLeft}(y) : \mathbf{type}}\,\to F}{\lambda e.\mathbf{isLeft}(x) \to \mathbf{isLeft}(y) : x = y \to \mathbf{type}}\,\Pi I}{\lambda y.\lambda e.\mathbf{isLeft}(x) \to \mathbf{isLeft}(y) : (y : A \uplus B) \to x = y \to \mathbf{type}}\,\Pi I}{\lambda x.\lambda y.\lambda e.\mathbf{isLeft}(x) \to \mathbf{isLeft}(y) : (x : A \uplus B) \to (y : A \uplus B) \to x = y \to \mathbf{type}}\,\Pi I \quad \dfrac{\dfrac{\begin{array}{c}\overline{x : A \uplus B}^{\,3}\\ \vdots \\ \mathbf{isLeft}(x) : \mathbf{type}\end{array} \quad \dfrac{\overline{p : \mathbf{isLeft}(x)}^{\,2} \quad \overline{p : \mathbf{isLeft}(x)}^{\,2}}{p : \mathbf{isLeft}(x)}\,wk}{\lambda p.p : \mathbf{isLeft}(x) \to \mathbf{isLeft}(x)}\,\to I, 2}{\lambda x.\lambda p.p : (x : A \uplus B) \to \mathbf{isLeft}(x) \to \mathbf{isLeft}(x)}\,\Pi I, 3}{\mathsf{idpeel}(e, \lambda x.\lambda p.p) : \mathbf{isLeft}(\mathsf{inl}(z)) \to \mathbf{isLeft}(\mathsf{inr}(z'))}\,=E \quad \dfrac{\dfrac{}{\langle\rangle : \top}\,\top I}{\langle\rangle : \overline{\top}}\,\mathbf{type}F}{\mathsf{idpeel}(e, \lambda x.\lambda p.p)() : \overline{\bot}}\,\to E}{\lambda e.\mathsf{idpeel}(e, \lambda x.\lambda p.p)() : (\mathsf{inl}(z) = \mathsf{inr}(z')) \to \overline{\bot}}\,\Pi I, 1$$

□

Lemma 26:

Let $h' \overset{def}{\equiv}$

$$\lambda y.\mathsf{case}\ y\ \mathsf{of}\ (\lambda x.\mathsf{case}\ h(x)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)), \lambda x.\mathsf{inl}(\mathsf{inr}(x))) : \mathbf{M}_{\mathsf{suc}(l)} \to \mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n.$$

Then,

$$\frac{l : \mathbf{Nat} \quad m : \mathbf{Nat} \quad n : \mathbf{Nat} \quad a : \mathbf{M}_l \quad b : \mathbf{M}_l \quad h : \mathbf{M}_l \to \mathbf{M}_m \uplus \mathbf{M}_n}{h'(\mathsf{inl}(a)) =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} h'(\mathsf{inl}(b)) \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b)\ true}$$

Proof.

$$h'(\mathsf{inl}(a)) \equiv_\beta \mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \tag{6.4}$$

$$h'(\mathsf{inl}(b)) \equiv_\beta \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \tag{6.5}$$

$$\dfrac{\begin{array}{ll} \dfrac{}{\begin{array}{l} h'(\mathsf{inl}(a)) \\ \equiv_\beta \mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \end{array}} & \dfrac{}{\begin{array}{l} h'(\mathsf{inl}(b)) \\ \equiv_\beta \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \end{array}} \\ \vdots\ \mathcal{D} & \vdots \\ XXX : & h'(\mathsf{inl}(a)) =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} h'(\mathsf{inl}(b)) \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \\ \mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) & \equiv_\beta \mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) & \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) & \quad \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \end{array}}{XXX : h'(\mathsf{inl}(a)) =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} h'(\mathsf{inl}(b)) \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b)}\ conv$$

$\underline{\mathcal{D}}$:

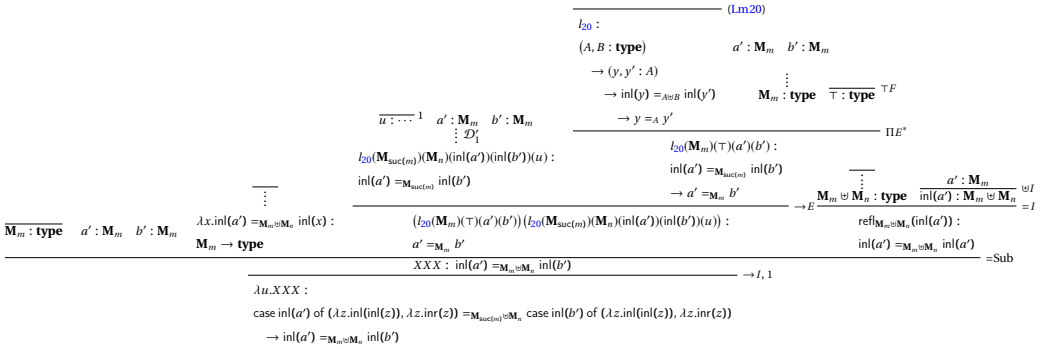
$$\dfrac{\dfrac{\dfrac{\dfrac{h : \ldots \quad b : \mathbf{M}_l}{\begin{array}{l} h(b) : \\ \mathbf{M}_m \uplus \mathbf{M}_n \end{array}} \to E \quad \begin{array}{l} xxx : \\ \mathsf{case}\ \mathsf{inl}(a')\ \mathsf{of}\ (\ldots, \ldots) \\ \quad = \mathsf{case}\ \mathsf{inl}(b')\ \mathsf{of}\ (\ldots, \ldots) \\ \to \mathsf{inl}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(b') \end{array} \quad \begin{array}{l} yyy : \\ \mathsf{case}\ \mathsf{inl}(a')\ \mathsf{of}\ (\ldots, \ldots) \\ \quad = \mathsf{case}\ \mathsf{inr}(b')\ \mathsf{of}\ (\ldots, \ldots) \\ \to \mathsf{inl}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(b') \end{array}}{\begin{array}{l} lll : \\ \mathsf{case}\ \mathsf{inl}(a')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \to \mathsf{inl}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \end{array}} \uplus E, 1 \quad \begin{array}{l} rrr : \\ \mathsf{case}\ \mathsf{inr}(a')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \to \mathsf{inr}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \end{array}}{XXX :} \uplus E, 3$$

$$\mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b)$$

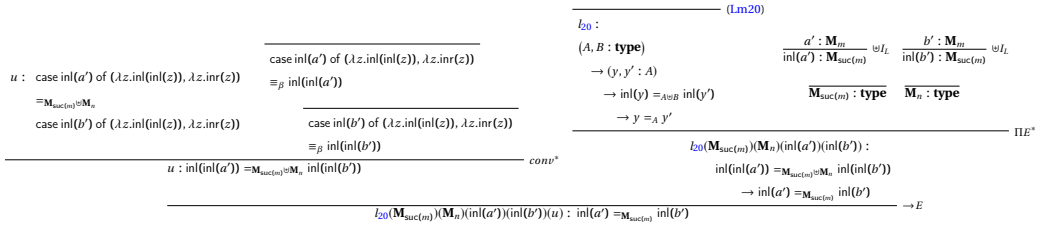
$$\dfrac{\dfrac{\dfrac{\dfrac{h : \ldots \quad b : \mathbf{M}_l}{\begin{array}{l} h(b) : \\ \mathbf{M}_m \uplus \mathbf{M}_n \end{array}} \to E \quad \begin{array}{l} xxx : \\ \mathsf{case}\ \mathsf{inr}(a')\ \mathsf{of}\ (\ldots, \ldots) \\ \quad = \mathsf{case}\ \mathsf{inl}(b')\ \mathsf{of}\ (\ldots, \ldots) \\ \to \mathsf{inr}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inl}(b') \end{array} \quad \begin{array}{l} yyy : \\ \mathsf{case}\ \mathsf{inr}(a')\ \mathsf{of}\ (\ldots, \ldots) \\ \quad = \mathsf{case}\ \mathsf{inr}(b')\ \mathsf{of}\ (\ldots, \ldots) \\ \to \mathsf{inr}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} \mathsf{inr}(b') \end{array}}{\begin{array}{l} lll : \\ \mathsf{case}\ \mathsf{inl}(a')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \to \mathsf{inl}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \end{array}} \uplus E, 1 \quad \begin{array}{l} rrr : \\ \mathsf{case}\ \mathsf{inr}(a')\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \quad =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \\ \to \mathsf{inr}(a') =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b) \end{array}}{XXX :} \uplus E, 3$$

$$\mathsf{case}\ h(a)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) =_{\mathbf{M}_{\mathsf{suc}(m)} \uplus \mathbf{M}_n} \mathsf{case}\ h(b)\ \mathsf{of}\ (\lambda z.\mathsf{inl}(\mathsf{inl}(z)), \lambda z.\mathsf{inr}(z)) \to h(a) =_{\mathbf{M}_m \uplus \mathbf{M}_n} h(b)$$

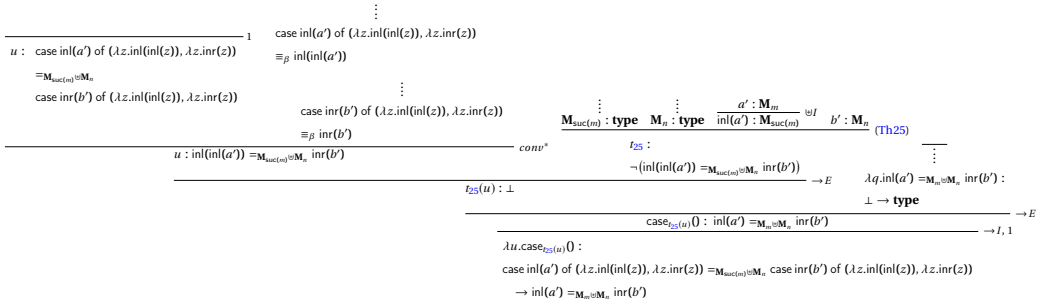
$\mathcal{D}_1$:



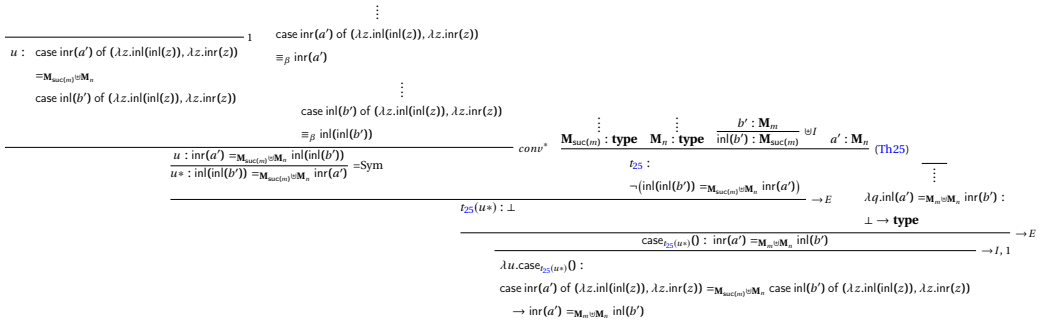
$\mathcal{D}'_1$:



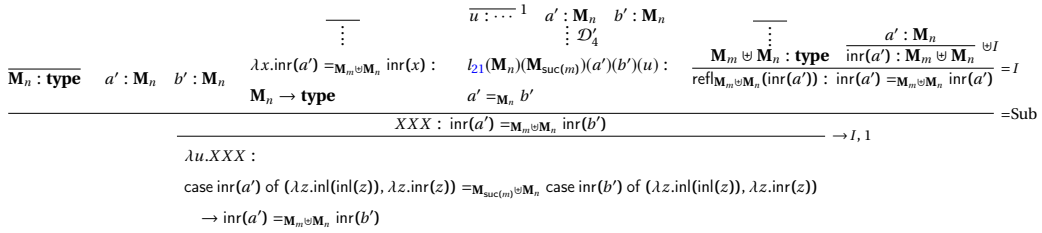
$\mathcal{D}_2$:



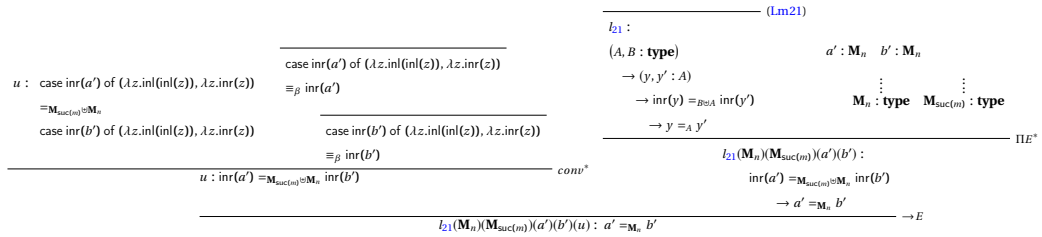
$\mathcal{D}_3$:



$\mathcal{D}_4$:



$\mathcal{D}'_4$:



□

Lemma 27: Let

$$\mathbf{i} \equiv \lambda k.\lambda k'.\lambda u.\mathsf{natrec}(\pi_1(u), \langle id_{\mathbf{M}_{k'}}, I_{id_{\mathbf{M}_{k'}}}\rangle, \lambda n'.\lambda q.\langle \lambda x.\mathsf{inl}((\pi q)(x)), I_{\lambda x.\mathsf{inl}((\pi q)(x))}\rangle)$$
$$: (k, k' : \mathbf{Nat}) \to (u : k' \leq k) \to \begin{bmatrix} f : \mathbf{M}_{k'} \to \mathbf{M}_k \\ \mathbf{injection}\,(i) \end{bmatrix}.$$

The following holds.

$$(k, l, n : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l) + n) \to (y : \mathbf{M}_l) \to$$
$$(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle n, p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle n, p_1\rangle))))(y)\ \mathit{true}$$

Proof. By Theorem 19, $\mathbf{p}_{19}(\langle n, p_1\rangle) \equiv_\beta \mathsf{suc}(\pi_1(\langle n, p_1\rangle)) \equiv_\beta \mathsf{suc}(n)$. Let $C_Y^X \overset{def}{\equiv} \mathsf{natrec}(X, \langle id_{\mathbf{M}_Y}, I_{id_{\mathbf{M}_Y}}\rangle, \lambda n'.\lambda q.\langle \lambda x.\mathsf{inl}((\pi q)(x)), I_{\lambda x.\mathsf{inl}((\pi q)(x))}\rangle)$. We have:

$$\mathbf{i}(k)(\mathsf{suc}(l))(\langle n, p_1\rangle) \equiv_\beta C^n_{\mathsf{suc}(l)}$$
$$\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle n, p_1\rangle)) \equiv_\beta C_l^{\mathsf{suc}(n)}.$$

The proof is by induction on n.

(1) For the case $n = 0$, we prove:

$$(k, l : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l) + 0) \to (y : \mathbf{M}_l) \to$$
$$(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle 0, p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle 0, p_1\rangle))))(y) \tag{6.6}$$

Assume $k, l : \mathbf{Nat}$, $p_1 : k = \mathsf{suc}(l) + 0$, and $y : \mathbf{M}_l$.

$$\begin{aligned}
& (\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle 0, p_1\rangle)))(\mathsf{inl}(y)) \\
\equiv_\beta\ & (\pi_1 C^0_{\mathsf{suc}(l)})(\mathsf{inl}(y)) \\
\equiv_\beta\ & (\pi_1\langle id_{\mathbf{M}_{\mathsf{suc}(l)}}, I_{id_{\mathbf{M}_{\mathsf{suc}(l)}}}\rangle)(\mathsf{inl}(y)) \\
\equiv_\beta\ & id_{\mathbf{M}_{\mathsf{suc}(l)}}(\mathsf{inl}(y)) \\
\equiv_\beta\ & \mathsf{inl}(y)
\end{aligned}$$

$$\begin{aligned}
& (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle 0, p_1\rangle))))(y) \\
\equiv_\beta\ & (\pi_1 C_l^{\mathsf{suc}(0)})(y) \\
\equiv_\beta\ & (\pi_1\langle \lambda x.\mathsf{inl}((\pi_1 C_l^0)(x)), I_{\lambda x.\mathsf{inl}((\pi_1 u)(x))}\rangle)(y) \\
\equiv_\beta\ & (\pi_1\langle \lambda x.\mathsf{inl}((\pi_1\langle id_{\mathbf{M}_l}, I_{id_{\mathbf{M}_l}}\rangle)(x)), I_{\lambda x.\mathsf{inl}((\pi u)(x))}\rangle)(y) \\
\equiv_\beta\ & (\lambda x.\mathsf{inl}((\pi_1\langle id_{\mathbf{M}_l}, I_{id_{\mathbf{M}_k}}\rangle)(x)))(y) \\
\equiv_\beta\ & \mathsf{inl}(id_{\mathbf{M}_l}(y)) \\
\equiv_\beta\ & \mathsf{inl}(y)
\end{aligned}$$

Hence, we have $(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle 0, p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle 0, p_1\rangle))))(y)$ and (6.6) holds.

(2) Assume that the formula holds for $n = m$ (IH).

$$\mathcal{IH} : (k, l : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l) + m) \to (y : \mathbf{M}_l) \to$$
$$(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle m, p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle m, p_1\rangle))))(y)$$

We will prove the formula for $\mathsf{suc}(m)$:

$$(k, l : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l) + \mathsf{suc}(m)) \to (y : \mathbf{M}_l) \to$$
$$(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle \mathsf{suc}(m), p_1 \rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle \mathsf{suc}(m), p_1 \rangle))))(y) \tag{6.7}$$

Assume $k, l : \mathbf{Nat},\ p_1 : k = \mathsf{suc}(l) + \mathsf{suc}(m)$ and $y : \mathbf{M}_l$. From IH,

$$\mathcal{IH}(\mathsf{suc}(l) + m)(l)(\mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(l) + m))(y) :$$
$$(\pi_1(\mathbf{i}(\mathsf{suc}(l) + m)(\mathsf{suc}(l))(\langle m, \mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(l) + m)\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(\mathsf{suc}(l) + m)(l)(\mathbf{p}_{19}(\langle m, \mathsf{refl}_{\mathbf{Nat}}(\mathsf{suc}(l) + m)\rangle)))))(y)$$

holds, namely,

$$(\pi_1 C^m_{\mathsf{suc}(l)})(\mathsf{inl}(y)) =_{\mathbf{M}_{\mathsf{suc}(l)+m}} (\pi_1 C^{\mathsf{suc}(m)}_l)(y)$$

has a proof. Now,

$$\begin{aligned}
& (\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle \mathsf{suc}(m), p_1 \rangle)))(\mathsf{inl}(y)) \\
\equiv_\beta\ & (\pi_1 C^{\mathsf{suc}(m)}_{\mathsf{suc}(l)})(\mathsf{inl}(y)) \\
\equiv_\beta\ & (\pi_1 \langle \lambda x.\mathsf{inl}((\pi_1 C^m_{\mathsf{suc}(l)})(x)), I_{\lambda x.\mathsf{inl}((\pi_1 q)(x))} \rangle)(\mathsf{inl}(y)) \\
\equiv_\beta\ & (\lambda x.\mathsf{inl}((\pi_1 C^m_{\mathsf{suc}(l)})(x)))(\mathsf{inl}(y)) \\
\equiv_\beta\ & \mathsf{inl}((\pi_1 C^m_{\mathsf{suc}(l)})(\mathsf{inl}(y))) \\
\equiv_\beta\ & \mathsf{inl}((\pi_1 C^{\mathsf{suc}(m)}_l)(y))
\end{aligned}$$

$$\begin{aligned}
& (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle \mathsf{suc}(m), p_1 \rangle))))(y) \\
\equiv_\beta\ & (\pi_1 C^{\mathsf{suc}(\mathsf{suc}(m))}_l)(y) \\
\equiv_\beta\ & (\pi_1 \langle \lambda x.\mathsf{inl}((\pi_1 C^{\mathsf{suc}(m)}_l)(x)), I_{\lambda x.\mathsf{inl}((\pi_1 q)(x))} \rangle)(y) \\
\equiv_\beta\ & (\lambda x.\mathsf{inl}((\pi_1 C^{\mathsf{suc}(m)}_l)(x)))(y) \\
\equiv_\beta\ & \mathsf{inl}((\pi_1 C^{\mathsf{suc}(m)}_l)(y))
\end{aligned}$$

Hence $(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle \mathsf{suc}(m), p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle \mathsf{suc}(m), p_1\rangle))))(y)$ and (6.7) holds.

From (1) and (2), we conclude that

$(k, l, n : \mathbf{Nat}) \to (p_1 : k = \mathsf{suc}(l) + n) \to (y : \mathbf{M}_l) \to$

$(\pi_1(\mathbf{i}(k)(\mathsf{suc}(l))(\langle n, p_1\rangle)))(\mathsf{inl}(y)) =_{\mathbf{M}_k} (\pi_1(\mathbf{i}(k)(l)(\mathbf{p}_{19}(\langle n, p_1\rangle))))(y)\ \mathit{true}$

□

Theorem 28

$$\frac{M : \{x : A \mid B(x)\} \quad N : \{x : A \mid B(x)\} \quad \mathsf{i}_{B'}(\mathsf{o}_B(M)) =_{\{x:A \mid B'(x)\}} \mathsf{i}_{B'}(\mathsf{o}_B(N))}{M =_{\{x:A \mid B(x)\}} N}$$

Proof. From $\mathsf{i}_{B'}(\mathsf{o}_B(M)) : \{x : A \mid B'(x)\}$, we obtain

$$\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M))) : A \tag{6.8}$$

by $\{|\}E$. We also obtain

$$?(M) : B(\mathsf{o}_B(M)) \tag{6.9}$$

from M by $\{|\}E_?$, which is equivalent to

$$?(M) : B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))) \tag{6.10}$$

as $\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M))) \equiv_\beta \mathsf{o}_B(M)$ holds. Hence, from (6.8) and (6.10), we get

$$\mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))) : \{x : A \mid B(x)\}, \tag{6.11}$$

which gives

$$\mathsf{refl} : \mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))) =_{\{x:A \mid B(x)\}} \mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))). \tag{6.12}$$

Now, $\mathsf{i}_{B'}(\mathsf{o}_B(M)) =_{\{x:A \mid B'(x)\}} \mathsf{i}_{B'}(\mathsf{o}_B(N))$ holds by the assumption. Hence we get

$$\mathsf{refl} : \mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))) =_{\{x:A \mid B(x)\}} \mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(N)))) \tag{6.13}$$

by substitution. As $\mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(M)))) \equiv_\beta M$ and $\mathsf{i}_B(\mathsf{o}_{B'}(\mathsf{i}_{B'}(\mathsf{o}_B(N)))) \equiv_\beta N$ hold, we get $M =_{\{x:A|B(x)\}} N$.

End of proof.

Appendix C: Lexical entries.

Expression	CCG Category	Semantic Representation
a_{nom}	$(S/(S\backslash NP))/N$	$\lambda n \lambda v.(x : \mathbf{e}) \times (u : nx) \times vx$
$some_{nom}$	$(S/(S\backslash NP))/N$	$\lambda n \lambda v.(x : \mathbf{e}) \times (u : nx) \times vx$
$every_{nom}$	$(S/(S\backslash NP))/N$	$\lambda n \lambda v.(u : (x : \mathbf{e}) \times nx) \rightarrow v(\pi_1 u)$
$most_{nom}$	$(S/(S\backslash NP))/N$	$\lambda n \lambda v.\mathbf{Most}(n, v)$
the_{nom}	$(S/(S\backslash NP))/N$	$\lambda n.\lambda v.v(\pi_1(@_i :: (x : \mathbf{entity}) \times nx))$
a_{acc}	$((S\backslash NP)\backslash((S\backslash NP)/NP))/N$	$\lambda n \lambda v \lambda x.(y : \mathbf{e}) \times (u : ny) \times vyx$
if	$S/S/S$	$\lambda p \lambda q.(u : p) \rightarrow q$
who	$(N\backslash N)/(S\backslash NP)$	$\lambda v \lambda n \lambda x.(nx \times vx)$
man	N	$\lambda x.\mathbf{man}(x)$
boy	N	$\lambda x.\mathbf{boy}(x)$
present	N	$\lambda x.\mathbf{present}(x)$
farmer	N	$\lambda x.\mathbf{farmer}(x)$
donkey	N	$\lambda x.\mathbf{donkey}(x)$
sister	RN	$\lambda y \lambda x.\mathbf{sisterOf}(x, y)$
enter	$S\backslash NP$	$\lambda x.\mathbf{enter}(x)$
beat	$(S\backslash NP)/NP$	$\lambda y \lambda x.\mathbf{beat}(x, y)$
own	$(S\backslash NP)/NP$	$\lambda y \lambda x.\mathbf{own}(x, y)$
receive	$(S\backslash NP)/NP$	$\lambda y \lambda x.\mathbf{receive}(x, y)$
open	$(S\backslash NP)/NP$	$\lambda y \lambda x.\mathbf{open}(x, y)$
he, she, it	NP	$@_i :: \mathbf{e}$
his, her	NP/RN	$\lambda r.\pi_1(@_i :: (x : \mathbf{e}) \times r(x, @_j :: \mathbf{e}))$
young	N/N	$\lambda n \lambda x.(nx \times \mathbf{young}(x))$

CCG Category	Semantic Type
$(S/(S\backslash NP))/N$	$(\mathbf{e} \rightarrow \mathbf{t}) \rightarrow (\mathbf{e} \rightarrow \mathbf{t}) \rightarrow \mathbf{t}$
$((S\backslash NP)\backslash((S\backslash NP)/NP))/N$	$(\mathbf{e} \rightarrow \mathbf{t}) \rightarrow (\mathbf{e} \rightarrow \mathbf{type}) \rightarrow \mathbf{e} \rightarrow \mathbf{t}$
S/S/S	$\mathbf{t} \rightarrow \mathbf{t} \rightarrow \mathbf{t}$
$(N\backslash N)/(S\backslash NP)$	$(\mathbf{e} \rightarrow \mathbf{t}) \rightarrow (\mathbf{e}to\mathbf{t}) \rightarrow \mathbf{e} \rightarrow \mathbf{t}$
NP	$\mathbf{e}$
N	$\mathbf{e} \rightarrow \mathbf{t}$
$S\backslash NP$	$\mathbf{e} \rightarrow \mathbf{t}$
$(S\backslash NP)/NP$	$\mathbf{e} \rightarrow \mathbf{e} \rightarrow \mathbf{t}$
RN	$\mathbf{e} \rightarrow \mathbf{e} \rightarrow \mathbf{t}$
NP/RN	$(\mathbf{e} \rightarrow \mathbf{e} \rightarrow \mathbf{t}) \rightarrow \mathbf{e}$

Bibliography

Lasha Abzianidze. LangPro: Natural language theorem prover. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 115–120, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Peter Aczel. The type theoretic interpretation of constructive set theory. In Logic Colloquium '77, pages 55–66. North-Holland Publishing Company, 1978.

Peter Aczel. The Type Theoretic Interpretation of Constructive Set Theory: Choice Principles. In A. S. Troelstra and D. van Dalen, editors, The L.E.J Brouwer Centenary Symbosium, Amsterdam, 1982.

Peter Aczel. The type theoretic interpretation of constructive set theory: Inductive definitions. In Barcan Marcus et al., editor, Logic, Methodology and Philosophy of Science, volume VII, pages 17–49. Elsevier, 1986.

Gabor Angeli and Christopher D. Manning. NaturalLI: Natural logic inference for common sense reasoning. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing

(EMNLP), pages 534–545, Doha, Qatar, October 2014. Association for Computational Linguistics.

Nicholas Asher and Zhaohui Luo. Formalisation of coercions in lexical semantics. In Sinn und Bedeutung 17, pages 63–80, 2012.

Nicholas Asher and Sylvain Pogodalla. A Montagovian Treatment of Modal Subordination. In Proceedings of SALT, volume 20, 2011.

H. P. Barendregt. Lambda Calculi with Types. In Handbook of logic in computer science, volume II, pages 117–309. Oxford University Press, 1992.

Jon Barwise and Robin Cooper. Barwise and Cooper 1981.pdf. In Linguistics and philosophy, Vol. 4, No. 2, pages 159–219. Springer, 1981.

David Ian Beaver. Presupposition. In Johan van Benthem and Alice ter Meulen, editors, Handbook of Logic and Language, pages 939–1008. North-Holland, Amsterdam, 1997.

Daisuke Bekki. Representing Anaphora with Dependent Types. In Nicholas Asher and S Soloviev, editors, Logical Aspects of Computational Linguistics, pages 14–29. Springer Berling Heidelberg, 2014.

Daisuke Bekki and Nicholas Asher. Logical polysemy and subtyping. In Yoichi Motomura, Alastair Butler, and Daisuke Bekki, editors, New Frontiers in Artificial Intelligence, volume 7856 of Lecture Notes in Computer Science, pages 17–24. Springer, 2013.

Daisuke Bekki and Elin McCready. CI via DTS. In Tsuyoshi Murata, Koji Mineshima, and Daisuke Bekki, editors, New Frontiers in Artificial

Intelligence: JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27-28, 2014, Revised Selected Papers, volume 9067, pages 23–36. Springer, 2015.

Daisuke Bekki and Koji Mineshima. Context-passing and underspecification in Dependent Type Semantics. In Stergios Chatzikyriakidis and Zhaohui Luo, editors, Modern Perspectives in Type Theoretical Semantics, Studies in Linguistics and Philosophy, pages 11–41. Springer, 2017.

Daisuke Bekki and Miho Satoh. Calculating projections via type checking. In Proceedings of TYpe Theory and LExical Semantics (TYTLES) in ESSLLI2015, Barcelona, Spain, 2015.

Johan Van Benthem. Meaning: Interpretation and inference. Synthese, 73(3):451–470, 1987.

Pascal Boldini. The reference of mass terms from a type theoretical point of view. In Proceedings of Forth International Workshop on Computational Semantics, 2001.

Adrian Brasoveanu. Donkey pluralities: plural information states versus non-atomic individuals. Linguistics and Philosophy, 31(2):129–209, 2008.

Lucas Champollion, Dylan Bumford, and Robert Henderson. Donkeys under discussion. Semantics and Pragmatics, 12:1–50, 2019.

Stergios Chatzikyriakidis and Zhaohui Luo. An account of natural language coordination in type theory with coercive subtyping. In Constraint Solving and Language Processing, LNCS 8114, pages 31–51. Springer, 2013.

Stergios Chatzikyriakidis and Zhaohui Luo. Natural Language Reasoning Using Proof-assistant Technology : Rich Typing and Beyond. In Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics (TTNLS), pages 37–45, Gothenburg, Sweden, 2014.

Stergios Chatzikyriakidis and Zhaohui Luo. On the Interpretation of Common Nouns: Types Versus Predicates, pages 43–70. Springer, 2017.

Stergios Chatzikyriakidis and Zhaohui Luo. Identity criteria of common nouns and dot-types for copredication. Approaches to Coercion and Polysemy, Oslo Studies in Language, 10(2), 2018.

Gennaro Chierchia. Anaphora and Dynamic Binding. Linguistics and philosophy, 15:111–183, 1992.

Herbert H. Clark. Bridging. In R. C. Schank and B. L. Nash-Webber, editors, Theoretical issues in natural language processing, pages 169–174. New York: Association for Computing Machinery, 1975.

Robin Cooper. Type theory and semantics in flux. Handbook of the Philosophy of Science, 14(2012):271–323, 2012.

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

G Evans. Pronouns. Linguistic Inquiry, 11(2):337–362, 1980.

N. Francez and G. Ben-Avi. Proof-Theoretic Reconstruction of Generalized Quantifiers. Journal of Semantics, pages 1–59, 2014.

Peter Thomas Geach. Reference and Generality: An Examination of Some Medieval and Modern Theories. Ithaca, NY: Cornell University Press, 1962.

Bart Geurts. Presuppositions and Pronouns. Elsevier, 1999. ISBN 1865843830.

Bart Geurts. Donkey business. Linguistics and Philosophy, 25:129–156, 2002.

Jeroen Groenendijk and Martin Stokhof. Dynamic predicate logic. Linguistics and Philosophy, 14(1):39–100, 1991.

Justyna Grudzińska and Marek Zawadowski. Generalized Quantifiers on Dependent Types: A System for Anaphora, pages 95–131. Springer International Publishing, Cham, 2017.

Martin Hackl. On the grammar and processing of proportional quantifiers: most versus more than half. Natural Language Semantics, 17:63–98, 2009.

Irene Heim. The Semantics of Definite and Indefinite Noun Phrases. PhD thesis, University of Massachusetts, Amherst, 1982.

Irene Heim. On the projection problem for presuppositions. In M. Barlow, D. Flickinger, and M. Wescoat, editors, Proceedings of WCCFL 2, pages 114–125. Stanford University, Stanford, CA, 1983.

Irene Heim. E-type pronouns and donkey anaphora. Linguistics and Philosophy, 13(2):137–177, 1990.

Jaakko Hintikka and Lauri Carlson. Conditionals, generic quantifiers, and other applications of subgames. In Esa Saarinen, editor, Game-Theoretical Semantics, pages 179–214. Springer, 1979.

Tim Hunter, Justin Halberda, Jeffrey Lidz, and Paul Pietroski. Beyond truth conditions: The semantics of "most". In Semantics and Linguistic Theory, volume 18, pages 447–464, 2008.

Thomas F. Icard and Lawrence S. Moss. Recent progress on monotonicity. In Linguistic Issues in Language Technology, Volume 9, 2014 - Perspectives on Semantic Representations for Textual Inference. CSLI Publications, 2014.

Bart Jacobs. Categorical logic and type theory, volume 141 of Studies in Logic and the Foundations of Mathematics. Elsevier, 1998.

Nirit Kadmon. On Unique and Non-unique Reference and Asymmetric Quantification, 1987.

Nirit Kadmon. Uniqueness. Linguistics and Philosophy, 13:273–324, 1990.

Hans Kamp and Uwe Reyle. From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory, volume 1. Kluwer Academic Publisher, 1993. ISBN 0792310284.

Hans Kamp, Josef van Genabith, and Uwe Reyle. Discourse representation theory. In Handbook of Philosophical Logic, volume 15, pages 125–394. Springer, 2011.

Makoto Kanazawa. Dynamic generalized quantifiers and monotonicity. In Makoto Kanazawa and Christopher J. Pi nón, editors, Dynamics, Polarity, and Quantification. CSLI publication, 1993.

Makoto Kanazawa. Weak vs. strong readings of donkey sentences and monotonicity inference in a dynamic setting. Linguistics and Philosophy, 2(17):109–158, 1994.

Makoto Kanazawa. Singular donkey pronouns are semantically singular. Linguistics and Philosophy, 24:383–403, 2001.

Lauri Karttunen. Discourse referents. In J. D. McCawley, editor, Syntax and Semantics Vol. 7, pages 363–386. Academic Press, 1976.

Jeffrey C King. Context dependent quantifiers and donkey anaphora. Canadian Journal of Philosophy, 34(sup1):97–127, 2004.

Eriko Kinoshita, Koji Mineshima, and Daisuke Bekki. An analysis of selectional restrictions with Dependent Type Semantics. In Proceedings of the 13th International Workshop on Logic and Engineering of Natural Language Semantics (LENLS13), pages 100–113, 2016.

Eriko Kinoshita, Koji Mineshima, and Daisuke Bekki. An analysis of selectional restrictions with Dependent Type Semantics. In Setsuya Kurahashi, Yuiko Ohta, Sachiyo Arai, Ken Satoh, and Daisuke Bekki, editors, New Frontiers in Artificial Intelligence: JSAI-isAI 2016. Lecture Notes in Computer Science, volume 10247, pages 19–32. Springer, 2017.

Eriko Kinoshita, Koji Mineshima, and Daisuke Bekki. Coercion as proof search in dependent type semantics. Approaches to Coercion and Polysemy, Oslo Studies in Language, 10(2):143–162, 2018.

Emiel Krahmer and Paul Piwek. Presupposition projection as proof construction. In Harry Bunt and Reinhard Muskens, editors, Computing Meaning, Studies in Linguistics & Philosophy, pages 281–300. Kluwer Academic Publishers, 1999.

Manfred Krifka. Parametrized sum individuals for plural anaphora. Linguistics and Philosophy, 19(6):555–598, 1996.

Yusuke Kubota and Robert Levine. Scope parallelism in coordination in Dependent Type Semantics. In Proceedings of the Twelfth International Workshop of Logic and Engineering of Natural Language Semantics (LENLS12), pages 149–162, 2015.

Yusuke Kubota, Koji Mineshima, Bob Levine, and Daisuke Bekki. Underspecification and interpretive parallelism in dependent type semantics. In Proceedings of the IWCS 2019 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 1–9, 2019.

David Lewis. Adverbs of quantification, pages 3âĂŞ–15. Cambridge University Press, 1975.

Per Lindström. First order predicate logic with generalized quantifiers. Theoria, 32(3):186–195, 1966.

Godehard Link. The Logical Analysis of Plurals and Mass Terms: A Lattice-Theoretical Approach. de Gruyter, 1983.

Godehard Link. Generalized Quantifiers and Plurals, pages 151–180. Springer, 1987.

Georgiana E. Lungu and Zhaohui Luo. Monotonicity reasoning in formal semantics based on modern type theories. In Nicholas Asher and Sergei

Soloviev, editors, Logical Aspects of Computational Linguistics, pages 138–148. Springer, 2014.

Zhaohui Luo. Common nouns as types. In Denis Béchet and Alexander Dikovsky, editors, Logical Aspects of Computational Linguistics, volume 7351 of Lecture Notes in Computer Science, pages 173–185. Springer, 2012a.

Zhaohui Luo. Formal semantics in modern type theories with coercive subtyping. Linguistics and Philosophy, 35(6):491–513, 2012b.

Zhaohui Luo. Proof Irrelevance in Type-Theoretical Semantics, pages 1–15. Springer, 2020.

Bill MacCartney and Christopher D. Manning. Modeling semantic containment and exclusion in natural language inference. In Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008), pages 521–528, Manchester, UK, August 2008. COLING 2008 Organizing Committee.

Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy Łoś, Helmut Pfeiffer, and Klaus-Peter Podewski, editors, Logic, Methodology and Philosophy of Science VI, volume 104 of Studies in Logic and the Foundations of Mathematics, pages 153–175. Elsevier, 1982.

Per Martin-Löf. Intuitionistic Type Theory. Bibliopolis Naples, 1984. ISBN 8870881059.

Gary Lee Milsark. Toward an explanation of certain peculiarities of the existential construction in English. PhD thesis, MIT, 1977.

Koji Mineshima. A presuppositional analysis of definite descriptions in proof theory. In Ken Satoh, Akihiro Inokuchi, Katashi Nagao, and Takahiro Kawamura, editors, New Frontiers in Artificial Intelligence: JSAI 2007 Conference and Workshops, Revised Selected Papers, volume 4914 of Lecture Notes in Computer Science, pages 214–227. Springer, 2008.

Koji Mineshima. Aspects of Inference in Natural Language. PhD thesis, Keio University, 2013.

Lawrence Moss. Natural Logic and Semantics. In Logic, Language and Meaning, pages 84–93. Springer Berling Heidelberg, 2010.

Andrzej Mostowski. On a generalization of quantifiers. Fundamenta Mathematicae, 44(2):12–36, 1957.

Bengt Nordström, Kent Petersson, and Jan M. Smith. Programming in Martin-Löf's Type Theory: An Introduction. Oxford University Press, 1990.

Rick Nouwen. Plural Pronominal Anaphora in Context : Dynamic Aspects of Quantification. PhD thesis, Utrecht Institute for Linguistics OTS, 2003.

Terence Parsons. Pronouns as paraphrases, 1978.

Rogelio Dávila Pérez. Semantics and Parsing in Intuitionistic Categorial Grammar. PhD thesis, University of Essex, 1995.

Stanley Peters and Dag Westerståhl. Quantifiers in Language and Logic. Clarendon Press, 2006.

Aarne Ranta. Type-theoretical Grammar. Oxford University Press, 1994.

Christian Retoré. The montagovian generative lexicon ΛTy_n: A type theoretical framework for natural language semantics. In Proceedings of TYPES 2013, pages 202–229, 2013.

Craige Roberts. Modal Subordination and Pronominal Anaphora in Discourse. Linguistics and Philosophy, 12:683–721, 1989.

Mark Steedman. The Syntactic Process. MIT Press/Bradford Books, 2000.

Göran Sundholm. Proof Theory and Meaning. In Handbook of Philosophical Logic, pages 471–506. Springer Netherlands, 1986.

Göran Sundholm. Constructive Generalized Quantifiers. Synthese, 79(1): 1–12, 1989.

Anna Szabolcsi. Quantification. Research Surveys in Linguistics. Cambridge University Press, 2010.

Mieszko Tałasiewicz. Donkey Philosophy. Evidence and Explanation in Theorizing about Semantics, pages 51–81. Peter Lang, Bern, Switzerland, 2018.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. Resolving modal anaphora in dependent type semantics. In Tsuyoshi Murata, Koji Mineshima, and Daisuke Bekki, editors, New Frontiers in Artificial Intelligence: JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27-28, 2014, Revised Selected Papers, volume 9067, pages 83–98. Springer, 2015.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. On the interpretation of dependent plural anaphora in a dependently-typed setting. In

Setsuya Kurahashi, Yuiko Ohta, Sachiyo Arai, Ken Satoh, and Daisuke Bekki, editors, New Frontiers in Artificial Intelligence, pages 123–137. Springer, 2017a.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. Factivity and presupposition in Dependent Type Semantics. Journal of Language Modelling, 5(2):385–420, 2017b.

Ribeka Tanaka, Koji Mineshima, and Daisuke Bekki. Paychecks, presupposition, and dependent types. In Proceedings of the Fifth Workshop on Natural Language and Computer Science (NLCS2018), Preprint no.215, Oxford, UK, 2018.

Selçuk Topal and Ahmet Çevik. Natural density and the quantifier "most". Journal of Logic, Language and Information, 29:511–523, 2020.

Johan van Benthem. A brief history of natural logic. In M Chakraborty, B Löwe, and M Nath Mitra, editors, Logic, Navya-Nyaya & Applications, Homage to Bimal Krishna Matilal. College Publications, London, 2008.

Martin van den Berg. Dynamic generalized quantifiers. In Jaap van Der Does and Jan van Eijck, editors, Quantifiers, Logic, and Language, pages 63–94. CSLI publications, 1996a.

Martin van den Berg. Some Aspects of the Internal Structure of Discourse: The Dynamics of Nominal Anaphora. PhD thesis, University of Amsterdam, 1996b.

Rob A. van der Sandt. Presupposition projection as anaphora resolution. Journal of Semantics, 9:333–377, 1992.

Rob A. van der Sandt and Bart Geurts. Presupposition, anaphora, and lexical content, pages 259–296. Springer, 1991.

Kazuki Watanabe, Koji Mineshima, and Daisuke Bekki. Questions in dependent type semantics. In Proceedings of the Sixth Workshop on Natural Language and Computer Science (NLCS'19), pages 23–33, Gothenburg, Sweden, 2019.

Narumi Watanabe, Daisuke Bekki, and Elin McCready. Japanese honorification: compositionality and expressivity. In S. Kawahara and M. Igarashi, editors, Proceedings of FAJL7: Formal Approaches to Japanese Linguistics, the MIT Working Papers in Linguistics 73, pages 265–276. 2014.

Yukiko Yana, Koji Mineshima, and Daisuke Bekki. Variable handling and compositionality: Comparing drt and dts. Journal of Logic, Language, and Information, 28:261–285, 2019.

Youngeun Yoon. Weak and Strong Interpretations of Quantifiers and Definite NPs in English and Korean. PhD thesis, University of Texas at Austin, 1994.

Youngeun Yoon. Total and partial predicates and the weak and strong interpretations. Natural Language Semantics, 4:217–236, 1996.