Translating a Fragment of Natural Deduction System for Natural Language into Modern Type Theory

Ivo Pezlar
The Czech Academy of Sciences, Institute of Philosophy
pezlar@flu.cas.cz

Abstract

In this paper, we investigate the possibility of translating a fragment of natural deduction system (NDS) for natural language semantics into modern type theory (MTT), originally suggested by Luo (2014). Our main goal will be to examine and translate the basic rules of NDS (namely, meta-rules, structural rules, identity rules, noun rules and rules for intersective and subsective adjectives) to MTT. Additionally, we will also consider some of their general features.

1 Introduction

In this paper, we will examine two proof-theoretic approaches to natural language semantics. Specifically, we will explore the possibility of embedding natural deduction systems (NDS, Francez 2015) into modern type theory (MTT, Chatzikyriakidis and Luo 2017b), originally hinted at by Luo (2014).

Our main goal will be to examine and try to translate the basic rules of NDS (namely, meta-rules, structural rules, identity rules, noun rules and rules for intersective and subsective adjectives) to MTT. Additionally, we will also consider some of their general features.

2 NDS and MTT: A Preliminary Overview

MTT is closely related to Martin-Löf's constructive type theory (Martin-Löf 1984) and it fully utilizes its rich type structures (dependent types, inductive types, ...). NDS is more similar to the standard logical approach based on Gentzen's natural deduction. In practice, this means that with NDS we are devising a purely proof-theoretic framework, but with MTT we are allowed more liberties due to its type-theoretic nature. This earned MTT some criticism from Francez, who regards MTT as 'a model-theoretic semantics, but one *constrained by proof-theoretic constraints*' (see Francez and Dyckhoff 2010, pp. 474–475). This point was not contested but rather embraced by Luo (see Luo 2014), who views his MTT as having both proof-theoretic and model-theoretic features (see Luo 2014, pp. 177–178).

The choice of a base system also dictates what will be the main vehicles for content: in MTT we work with judgements of the form $a : A$, where a is a so-called proof object (proof term, witness, justification, ...) and A a proposition/type, while in NDS we work with formulas (or (pseudo-)sentence in the case of natural language fragment). From a technical standpoint, probably the most important difference between judgements and formulas is that judgements have effective procedural content, i.e., they are decidable. More specifically, given a judgement $a : A$ we should be always able to compute whether a is an object of the type A. Consider e.g., the judgement $1 : Nat$, i.e., the judgement that 1 is a natural number. The number 1 is in constructive type theories usually defined simply as $s(0) : Nat$, i.e., as the successor of 0. This form alone tells us that 1 is indeed a natural number, since in MTT all natural numbers are either 0 or have the form $s(a)$ where a is a natural number.[1] This is not the case with

[1]This follows from the introduction rules for natural numbers. For a proper definition of natural numbers in MTT, see e.g., Martin-Löf (1984).

Proceedings of the IWCS 2019 Workshop on Computing Semantics with Types, Frames and Related Structures, pages 10–18
Gothenburg, Sweden, May 24, 2019.

formulas, which are generally undecidable. For example, suppose that $a \in P$ is a formula of predicate logic capturing the fact that a has some property P, more specifically that a is in the set P. Whether some element is a member of a set or not is, however, generally not decidable. Consequently, $a \in P$ is undecidable as well.[2] Other points of discord could be found as well. For example, if our translation of NDS into MTT succeeds, we lose some of the nice 'philosophical' properties of NDS (e.g., fewer ontological commitments).[3] On the other hand, if we are solely interested in formal semantics, this does not need to concern us.

So far we have discussed only the differences between NDS and MTT, but we can identify important similarities as well. The key intuition of proof-theoretic semantics that meanings are constituted via canonical proofs[4] is, of course, present in both systems. As Francez states:

> For compound sentences, sentential meanings are defined as the (contextualised) *collection of canonical derivations* [...]. This is very much in the spirit of the modern approach 'propositions as types' (for example, Martin-Löf 1984), the inhabitants of a type being the the proofs [...]. (Francez 2015, p. 46)

As expected, both systems rely on the standard scheme of introduction rules mirrored by the corresponding elimination rules. MTT adds to this mix, however, also formation rules and computation rules (also called equality rules), which can be understood as rules for assembling well-formed terms and for term reductions, respectively. Both frameworks also avoid the Fregean function-argument form of predication ($F(a)$) and move towards the more classical subject-predicate predication (S is P) reminiscent of Aristotelian logic. More specifically, NDS utilizes pseudo-sentences of the general form a isa A, while MTT operates with judgements of the general form $a : A$. NDS also relies on the so-called reification of meaning, which puts it apart from most PTS approaches, but closer to MTT:

> The approach I am proposing in this book is rather to conceive of PTS as providing an *explicit definition* of meanings by meaning-conferring rules. Thus, if ξ is some meaning bearing expression, PTS should provide some *proof-theoretic semantic value* of the form $[\xi] =^{df.} \cdots$ as the meaning of ξ. I refer to this semantic value as a reified meaning. (Francez 2015, p. 7)

This reification is similar to MTT and its underlying conception of propositions as sets of proof objects. Assume that we have two proof objects $\lambda x.x$ and $\lambda y.y$ for the proposition $A \supset A$. These two proof objects differ only in the names of bound variables, i.e., they are α-equivalent. In MTT, we can express all this as $\lambda x.x : A \supset A$, $\lambda y.y : A \supset A$, and finally $\lambda x.x =_\alpha \lambda y.y : A \supset A$. Compare this with NDS and its expression $[\![A \supset A]\!]^{Ic}$ denoting the set of all I-canonical proofs of $A \supset A$. Since $\lambda x.x$ and $\lambda y.y$ are essentially understood as reified proofs or codes for proofs, we can see that both $\lambda x.x =_\alpha \lambda y.y : A \supset A$ and $[\![A \supset A]\!]^{Ic}$ capture a similar intuition.

3 From NDS to MTT

In our translation, we start with meta-rules (3.1), then we consider identity rules (3.2), rules for proper names (3.3), and finally we will examine rules for intersective and subsective adjectives (3.4). As we shall see, all the discussed NDS rules can be embedded into MTT semantics and justified either as admissible rules or derivable rules.

The translation method we utilize is based on the suggestion made by Luo (2014). Generally speaking, the translation method has two steps: 1) identifying the suitable expressions of NDS for the application of translation function $[\![\ ]\!]$ (a syntactic step), and 2) finding the appropriate translations in MTT

[2] Many other distinctions between formulas and judgements can be identified, but since it is a topic beyond the scope of this paper, we will not pursue it further.

[3] See Francez (2015), p. 9.

[4] A proof is canonical if and only if it ends with an application of an introduction rule. We call such proof an I-canonical proof (see Def. 1.5.8, Francez 2015, p. 36). In the framework of MTT, this corresponds to proof objects being in a canonical form, i.e., a form given to them by the corresponding introduction rules.

(a semantic step). As an simple example, suppose we have an expression Alice isa student, which is a proper sentence of NDS, hence we can apply the translation function $[\![\text{Alice isa student}]\!]$. As the corresponding translation in MTT, we get the judgement $Alice : Student$. Although the translation is not always as straightforward as Luo's quote might suggest, we will show that in general it can be successfully deployed for all the basic rules (meta-rules, identity rules, noun rules, adjective rules).

3.1 Meta-Rules

The meta-rules for NDS (see below) are intended to confer meaning of sentences from the natural language fragment containing only in/transitive verbs, determiner phrases with a singular noun, determiners 'every' and 'some' and a copula 'is' (see Francez 2015). The rules for determiners come in pairs of introduction rules (I-rules) and (generalized) elimination rules (E-rules) and they behave in accordance with the standard intuitionistic explanations of the corresponding quantifiers.

$$\frac{}{\Gamma, S \vdash S}\,(Ax) \qquad \frac{\Gamma, \mathbf{j} \text{ isa } X \vdash S[\mathbf{j}]}{\Gamma \vdash S[(\text{every } X)]}\,(eI) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ isa } X \qquad \Gamma \vdash S[\mathbf{j}]}{\Gamma \vdash S[(\text{some } X)]}\,(sI)$$

$$\frac{\Gamma \vdash S[(\text{every } X)] \quad \Gamma \vdash \mathbf{j} \text{ isa } X \quad \Gamma, S[\mathbf{j}] \vdash S'}{\Gamma \vdash S'}\,(eE)$$

$$\frac{\Gamma \vdash S[(\text{some } X)] \quad \Gamma, \mathbf{j} \text{ isa } X, S[\mathbf{j}] \vdash S'}{\Gamma \vdash S'}\,(sE)$$

where $\mathbf{j}$ is fresh for $\Gamma, S[\text{every } X]$ in (eI), and for $S[\text{some } X], S'$ in (sE).[5]

First, some additional explanations are in order: $\mathbf{j}$, X, and S are meta-variables for individual parameters (determiner phrases, ...), nouns (including compound nouns), and (affirmative) pseudo-sentences,[6] respectively, while isa serves as a copula. Furthermore, expression of the form $S[\mathbf{j}]$ means that $\mathbf{j}$ occupies a determiner phrase position in the sentence S. every and some are determiners (for more, see Francez 2015).

First, we present all the translated variants for meta-rules, then we add comments and examples.

$$\frac{}{[\![\Gamma]\!], [\![S]\!] \vdash [\![S]\!]}\,(Ax)' \qquad \frac{[\![\Gamma]\!], j : [\![X]\!] \vdash [\![S[j]]\!]}{[\![\Gamma]\!] \vdash [\![S[\forall([\![X]\!])]]\!]}\,(eI)' \qquad \frac{[\![\Gamma]\!] \vdash j : [\![X]\!] \quad [\![\Gamma]\!] \vdash [\![S[j]]\!]}{[\![\Gamma]\!] \vdash [\![S[\exists([\![X]\!])]]\!]}\,(sI)'$$

$$\frac{\Gamma \vdash [\![S[\forall([\![X]\!])]]\!] \quad [\![\Gamma]\!] \vdash j : [\![X]\!] \quad [\![\Gamma]\!], [\![S[j]]\!] \vdash [\![S']\!]}{[\![\Gamma]\!] \vdash [\![S']\!]}\,(eE)'$$

$$\frac{[\![\Gamma]\!] \vdash [\![S[\exists([\![X]\!])]]\!] \quad [\![\Gamma]\!], j : [\![X]\!], [\![S[j]]\!] \vdash [\![S']\!]}{[\![\Gamma]\!] \vdash [\![S']\!]}\,(sE)'$$

Comments. The rule $(Ax)'$ is justified by the structural rule assump from MTT. The rules $(eI)'$ and $(sI)'$ are justified by the rules Π-intro and Σ-intro (or more precisely, via $\forall$-intro and $\exists$-intro that are based upon them), respectively. Analogously for the rules $(eE)'$ and $(sE)'$. The context Γ from NDS, i.e., a finite list of formulas, is translated into a list of judgements. More specifically, in MTT, $\Gamma \vdash a : A$ is a hypothetical judgement properly unpacked as $a_1 : A, \ldots, a_n : A \vdash a : A$ where n is the number of assumptions in the context. The copula 'isa' is used for predication in NDS. In MTT, predication is achieved with the use of colon ':', so translating $\mathbf{j}$ isa X as $j : [\![X]\!]$ seems as a good fit. This decision dictates the rest of the translation: if we replace isa with :, then the left-hand side has to be some object and the right-hand side has to be its type. The most straightforward way to treat the determiner every

[5] Since we will not be interested here in the issue of quantifier scope ambiguity, we omit the corresponding explicit scope indicators from the rules. For example, the rule (sI) in its fully disclosed variant looks like $\frac{\Gamma, \mathbf{j} \text{ isa } X \vdash S[\mathbf{j}]}{\Gamma \vdash S[(\text{every } X)_{r(S[\mathbf{j}]+1)}]}\,(eI)$.

[6] A pseudo-sentence (of the object language) is a schematic sentences with occurrences of at least one parameter, e.g., $\mathbf{j}$ isa X. Example of a (pseudo-)sentence might be e.g., $\mathbf{j}$ isa student.

seems to be simply to take it as the universal quantifier $\forall$, which is defined in MTT via the Π type.[7] In other words, we will capture $S[(\text{every } X)]$ as sentential function over individual parameters, i.e., as an indexed family of types over the objects of type X. Analogously for the determiner some that can be treated via the Σ type.[8]

Examples. The following derivation from NDS:

$$\frac{\Gamma, \mathbf{j} \text{ isa girl} \vdash \mathbf{j} \text{ smiles}}{\Gamma \vdash \text{every girl smiles}}\,(eI)$$

gets as its MTT variant the following derivation:

$$\frac{\Gamma, j : Girl \vdash s(j) : Smiles(j)}{\Gamma \vdash \lambda j.s(j) : (\forall j : Girl)Smiles(j)}\,(eI)'$$

Note that in MTT, the noun girl is captured as the type $Girl$ and the predicate smiles as the dependent type $Smiles(j)$. Furthermore, note that the relationship between $\mathbf{j}$ and S in NDS, i.e., $S[\mathbf{j}]$, is captured in MTT by interpreting $S[\mathbf{j}]$ as a type of sentence (proposition) depending on the assumption $j : Girl$. We can also see that this formalization is in accord with Francez's own approach:

> A proof of $S[(\text{every } X)]$ is a function mapping each proof of $\mathbf{j}$ isa X (for an arbitrary fresh parameter $\mathbf{j}$) into a proof of $S[\mathbf{j}]$. (Francez 2015, p. 247)

On our approach, the proof of $(\forall j : Girl)Smiles(j)$ is the proof object $\lambda j.s(j)$ which is a function (or rather a function name) that takes a proof object j and returns a proof object $s(j)$.[9]

As a more complicated example with a transitive verb, the NDS derivation:

$$\frac{\mathbf{k} \text{ isa boy} \qquad \dfrac{\Gamma, \mathbf{j} \text{ isa girl} \vdash \mathbf{j} \text{ loves } \mathbf{k}}{\Gamma \vdash \text{every girl loves } \mathbf{k}}\,(eI)}{\Gamma \vdash \text{every girl loves some boy}}\,(sI)$$

becomes:
$$\frac{k : Boy \qquad \dfrac{\Gamma, j : Girl, y : A \vdash l(j,y) : Loves(j,y)}{\Gamma, y : A \vdash \lambda j.l(j,y) : (\forall j : Girl)Loves(j,y)}\,(eI)'}{\Gamma \vdash (k, \lambda j.l(j,y)) : (\exists k : Boy)(\forall j : Girl)Loves(j,k)}\,(sI)'$$

And a derivations such as: $\dfrac{\Gamma, \mathbf{k} \text{ isa boy} \vdash \mathbf{j} \text{ loves } \mathbf{k}}{\Gamma \vdash \mathbf{j} \text{ loves every boy}}\,(eI)$ becomes:

$$\frac{\Gamma, k : Boy, x : A \vdash l(x,k) : Loves(x,k)}{\Gamma, x : A \vdash \lambda k.l(x,k) : (\forall k : Boy)Loves(x,k)}\,(eI)'\,.$$

3.2 Identity Rules

In the natural language fragment, Francez works with a set of rules determining the behaviour of the copula is, which is treated as 'a disguised identity' (Francez 2015, p. 250). Naturally, it behaves in the same way. The collection of rules is as follows:[10]

$$\frac{\Gamma, S[\mathbf{j}] \vdash S[\mathbf{k}]}{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{k}}\,(isI) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{k} \quad \Gamma \vdash S[\mathbf{j}]}{\Gamma \vdash S[\mathbf{k}]}\,(is\hat{E}) \qquad \frac{}{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{j}}\,(is/r) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{k}}{\Gamma \vdash \mathbf{k} \text{ is } \mathbf{j}}\,(is/s)$$

[7] Π type is essentially a Cartesian product of a family of sets.

[8] NDS also utilizes the structural rules of contraction and weakening, which correspond to multiple and vacuous discharge of assumptions in MTT.

[9] Note that since we are capturing $\mathbf{j}$ isa X as a judgement and not a proposition, it would make no sense speaking about its proof objects.

[10] We skip over the generalized variant (isE) (see Francez 2015, p. 250) and use only the derived version $(is\hat{E})$ presented above.

$$\frac{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{k} \qquad \Gamma \vdash \mathbf{k} \text{ is } \mathbf{l}}{\Gamma \vdash \mathbf{j} \text{ is } \mathbf{l}}\ (is/t)$$

Before we approach the translation of these rules, we have to address that in MTT there are two kinds of identity: *propositional* (or extensional) and *judgemental* (intensional, definitional). Probably the most important difference is that the judgemental identity, represented as $a = b : A$, is decidable, while the propositional identity, usually written as $Id(A, a, b)$, is not.[11] So what type of identity describe the rules above? If their were describing judgemental identity, then the translation of the reflexivity, symmetry and transitivity would be straightforward. For example, the MTT variant of $(is - sym)$ would be: $\frac{a = b : A}{b = a : A}$ symm , etc. However, given the fact that the identity rules of NDS have dedicated I- and E-rules, they seem to be describing propositional identity. In MTT, we cannot *introduce* judgemental identity in a way we would introduce e.g., some logical operator.[12]

The translated variants would be:

$$\frac{[\![\Gamma]\!], j = k : A, [\![S[j]]\!] \vdash [\![S[k]]\!]}{[\![\Gamma]\!] \vdash Id(A, j, k)}\ (isI)' \qquad \frac{[\![\Gamma]\!] \vdash Id(A, j, k) \qquad [\![\Gamma]\!] \vdash [\![S[j]]\!]}{[\![\Gamma]\!] \vdash [\![S[k]]\!]}\ (is\hat{E})'$$

$$\frac{}{[\![\Gamma]\!] \vdash Id(A, j, j)}\ (is/r)' \qquad \frac{[\![\Gamma]\!] \vdash Id(A, j, k)}{[\![\Gamma]\!] \vdash Id(A, k, j)}\ (is/s)'$$

$$\frac{[\![\Gamma]\!] \vdash Id(A, j, k) \qquad [\![\Gamma]\!] \vdash Id(A, k, l)}{[\![\Gamma]\!] \vdash Id(A, j, l)}\ (is/t)'$$

where A is the type of the objects j and k that we would use to represent the individual parameters. For example, if we have $\mathbf{j}$ isa girl in NDS, then we assume that $j : Girl$ in MTT.

Comments. Validity of the $(isI)'$ rule follows from the fact that, in general, from $a = b : A$ we can deduce $refl(A, a) : Id(A, a, b)$, which can be derived as a rule from Id-intro using substitution and set equality rules:

$$\frac{\dfrac{\Gamma, x : A \vdash Id(A, a, x) : type \qquad \Gamma \vdash a = b : A}{\Gamma \vdash Id(A, a, a) = Id(A, a, b)} \qquad \dfrac{\Gamma \vdash a : A}{\Gamma \vdash refl(A, a) : Id(A, a, a)}}{\Gamma \vdash refl(A, a) : Id(A, a, b)}$$

The rule $(is - refl)'$ is justified by the Id-intro rule. The rule $(is\hat{E})'$ is sanctioned by the Id-elim rule. With the constant $refl$ provided by the Id-intro rule, we can also prove symmetry and transitivity of the relation Id and supply the corresponding derived rules for symmetry:[13]

$$\frac{d : Id(A, a, b)}{symm(d) : Id(A, b, a)} \qquad \frac{d : Id(A, a, b) \qquad e : Id(A, b, c)}{trans(d, e) : Id(A, a, c)}$$

which in turn justify the rules $(is/s)'$ and $(is/t)'$, respectively.

3.3 Proper Names Rules

The I- and E-rules for proper names, which are ranged over by meta-variables N, M, are specified as follows (see Francez 2015, p. 251):

$$\frac{\Gamma \vdash \mathbf{j} \text{ is } N \qquad \Gamma \vdash S[\mathbf{j}]}{\Gamma \vdash S[N]}\ (nI) \qquad \frac{\Gamma \vdash S[N] \qquad \Gamma, \mathbf{j} \text{ is } N, S[\mathbf{j}] \vdash S'}{\Gamma \vdash S'}\ (nE)$$

[11] As a concrete example of this distinction, consider e.g., the difference between numbers $5 + 7$ and 12. While their are both extensionally equal (they both denote the number 12), they are not intensionally equal (they do not denote it in the same way). Alternatively, we can view this distinction as a difference between abstract objects and linguistic terms.

[12] Recall that $a = b : A$ is one of the basic kinds of MTT judgements.

[13] For proofs and definitions of the constants $symm$ and $trans$, see e.g., Nordström et al. (1990).

The translated variants will become:

$$\frac{[\![\Gamma]\!] \vdash Id(A,j,n) \quad [\![\Gamma]\!] \vdash [\![S[j]]\!]}{[\![\Gamma]\!] \vdash [\![S[n]]\!]}\,(nI)' \qquad \frac{[\![\Gamma]\!] \vdash [\![S[n]]\!] \quad [\![\Gamma]\!], Id(A,j,n), [\![S[j]]\!] \vdash [\![S']\!]}{[\![\Gamma]\!] \vdash [\![S']\!]}\,(nE)'$$

Comments. Similarly to the case above, we capture **j** is N via propositional identity. Furthermore, note that $(nI)'$ is essentially just a special case of Id-elim rule from MTT. More generally, it is an instance of the Leibniz's principle of the indiscernibility of identicals, which can be in MTT expressed as follows:

$$(\forall x : A)(\forall y : A)(\forall p : Id(A,x,y))B(x) \supset B(y).$$

Informally, it states that whenever we have two identical names, we can freely swap them in any sentence they appear. For its proof using the Id-elim rule, see e.g., Martin-Löf (1984). The $(nE)'$ rule is also justifiable via Id-elim.

Examples. As an example, we construct the derivation $(8.3.35)$ (see Francez 2015, p. 251) establishing that:

$$\text{Rachel isa girl, every girl smiles} \vdash \text{Rachel smiles}$$

which under our translation amounts to:

$$Rachel : Girl, \lambda x.s(x) : (\forall x : Girl)Smiles(x) \vdash s(Rachel) : Smiles(Rachel)$$

It can be derived as follows:

$$\dfrac{\dfrac{Rachel : Girl \qquad \dfrac{\dfrac{Id(Girl,r,Rachel) \qquad \dfrac{r : Girl \qquad \lambda x.s(x) : (\forall x : Girl)Smiles(x)}{s(r) : Smiles(r)}\,(e\hat{E})'}{s(Rachel) : Smiles(Rachel)}\,(nI)'}{s(Rachel) : Smiles(Rachel)}\,(nE)'}{}$$

3.4 Adjectives

3.4.1 Intersective Adjectives

In this section, we examine and compare NDS and MTT approaches to intersective adjectives. We will take intersective adjectives as specified by adhering to the following two kinds of rules:

$$\frac{a \text{ is } Adj\ Noun}{a \text{ is } Adj}\,intA_1 \qquad \frac{a \text{ is } Adj\ Noun}{a \text{ is } Noun}\,intA_2$$

For example,

$$\frac{a \text{ is } black\ car}{a \text{ is } black}\,intA_1 \qquad \frac{a \text{ is } black\ car}{a \text{ is } car}\,intA_2$$

Hence, intersective adjectives are those adjectives that allow inferring from 'a is $Adj\ Noun$' that the underlying object of predication a possesses both its constituents separately: the noun $Noun$, i.e., 'a is $Noun$', as well as the intersective adjective Adj, i.e., 'a is Adj'. Observe that the compound $Adj\ Noun$ of intersective adjective and noun behaves similarly to the logical connective conjunction in standard natural deduction. More specifically, in natural deduction, conjunction has associated two elimination rules: $\frac{A \wedge B}{A}\,\wedge_1$-E and $\frac{A \wedge B}{B}\,\wedge_2$-E . These two rules correspond in their behaviour to rules $intA_1$ and $intA_1$. Elimination rules for conjunction allow deducing both its conjuncts A and B separately, and, analogously, rules for intersective adjectives allow deducing both its parts Adj and $Noun$. Hence, we could say that intersective adjectives preserve inferential content. We will utilize this fact later.

In NDS, intersective adjectives appear within *ground* pseudo-sentences of the form[14] **j** is A where A is a meta-variable for intersective adjective (see above). The corresponding rules are:

[14]Ground pseudo-sentences are pseudo-sentences that contain only parameters in every determiner phrase position. For example, **j** smiles. For more about ground pseudo-sentences, see Francez (2015), p. 245.

$$\frac{\Gamma \vdash \mathbf{j} \text{ isa } X \qquad \Gamma \vdash \mathbf{j} \text{ is } A}{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}\,(adjI) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ isa } A\ X \qquad \Gamma, \mathbf{j} \text{ isa } X, \mathbf{j} \text{ is } A \vdash S'}{\Gamma \vdash S'}\,(adjE)$$

From the rule $(adjE)$ we can obtain the following derived rules (see Francez 2015, p. 252):

$$\frac{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}{\Gamma \vdash \mathbf{j} \text{ isa } X}\,(adj\hat{E}_1) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}{\Gamma \vdash \mathbf{j} \text{ is } A}\,(adj\hat{E}_2)$$

It is easy to check that these two rules correspond to our general rules $intA_1$, $intA_2$ for intersective adjective and, consequently, to conjunction elimination rules. Furthermore, it now becomes clear that the original rule $(adjE)$ corresponds to the generalized conjunction elimination rule (see e.g., Negri et al. 2001): $\dfrac{\begin{matrix}A \wedge B \\ \\ \end{matrix} \qquad \begin{matrix}[A \wedge B] \\ C \end{matrix}}{C}\,\wedge\text{-GE}$. Before we get to the translation of the above rules, we will first discuss how adjectives are treated in MTT. Intersective adjectives are generally analyzed with Σ type, i.e., the same type that is also used for defining conjunction.[15] For example, the expression 'black car' would be captured as the type: $(\Sigma x : Car)Black(x)$, i.e., the type of cars that are black ($Black(x)$ is considered as a property/propositional function). The corresponding proof object is a pair (x, y) such that $x : Car$ and $y : Black(x)$, i.e., y is a justification (proof object) that x is black. Hence, common nouns are interpreted as distinct types, so we will get types of cars, animals, humans, etc. (so-called many-sorted type theory).[16]

The corresponding MTT introduction for $(adjI)$ would then be:[17]

$$\frac{\Gamma \vdash n : Noun \qquad \Gamma \vdash a : intAdj(x)}{\Gamma \vdash (n, a) : (\Sigma x : Noun)intAdj(x)}$$

We mentioned above that intersective adjectives should be conservative with respect to their inferential content. We can test this with projection functions fst and snd. Intuitively, fst and snd are operations that return the first and the second element of the proof object of the pair type $(\Sigma x : Noun)intAdj(x)$, respectively. For example, assume that we have the proof object p such that $p = (x, y)$ of $(\Sigma x : Car)Black(x)$, then $fst(p) = x : Car$ and $snd(p) = y : Black(fst(p))$.[18] The corresponding elimination rules will then be as follows:

$$\frac{\Gamma \vdash c : (\Sigma x : Noun)intAdj(x)}{fst(c) : Noun}\,(adj\hat{E}_1)' \qquad \frac{\Gamma \vdash c : (\Sigma x : Noun)intAdj(x)}{snd(c) : intAdj(x)}\,(adj\hat{E}_2)'$$

The projections fst and snd can be defined using the non-canonical constant E (brought by Σ-elim rule) in the following manner: $fst(c) = E(c, (x, y)x)$ and $snd(c) = E(c, (x, y)y)$, respectively.

Now, we can finally get to the translation of the rules themselves (we skip the generalized elimination variant):

$$\frac{[\![\Gamma]\!] \vdash j : [\![X]\!] \quad [\![\Gamma]\!] \vdash k : [\![A]\!]}{[\![\Gamma]\!] \vdash (j, k) : [\![A\ X]\!]}\,(adjI)' \quad \frac{[\![\Gamma]\!] \vdash j : [\![A\ X]\!]}{[\![\Gamma]\!] \vdash snd(j) : [\![A]\!]}\,(adj\hat{E}_1)' \qquad \frac{[\![\Gamma]\!] \vdash j : [\![A\ X]\!]}{[\![\Gamma]\!] \vdash fst(j) : [\![X]\!]}\,(adj\hat{E}_2)'$$

Comments. As discussed above, the rules for intersective adjectives are justified by the corresponding Σ type rules.

Examples. The following NDS derivation (see Francez 2015, p. 252): $\dfrac{\mathbf{j} \text{ isa } Y \qquad \dfrac{\mathbf{j} \text{ isa } A\ X}{\mathbf{j} \text{ is } A}\,(adj\hat{E}_2)}{\mathbf{j} \text{ isa } A\ Y}\,(adjI)$

becomes: $\dfrac{k : [\![Y]\!] \qquad \dfrac{j : [\![A\ X]\!]}{snd(j) : [\![A]\!]}\,(adj\hat{E}_2)'}{(k, snd(j)) : [\![A\ Y]\!]}\,(adjI)'$

[15] Compare with Ranta (1994), pp. 34–35. However, it is important to note that in order for this formalization to work, subtyping has to be adopted as well. See Luo (2012).

[16] Hence, e.g., a sentence 'Alice is a girl' will be understood as stating that Alice is an object of type girl, not as stating that predicate girl is applied to the individual Alice.

[17] Note that in MTT the relation between noun and adjective is handled via the Σ type, while in NDS it is achieved by the fact that both predications '$\mathsf{isa}\ X$' and '$\mathsf{is}\ A$' share the same parameter $\mathbf{j}$.

[18] As we can see, our initial rules $intA_1$ and $intA_2$ correspond to rules for left and right conjunction elimination defined via projections fst, snd, respectively.

3.4.2 Subsective Adjectives

We specify subsective adjectives by the following two rules:

$$\frac{a \text{ is } Adj\ Noun}{a \text{ is } Noun}\ subA_1 \qquad \frac{a \text{ is } Adj\ Noun}{a \text{ is } Adj_N}\ subA_2$$

For example,

$$\frac{a \text{ is } large\ mouse}{a \text{ is } mouse}\ intA_1 \qquad \frac{a \text{ is } large\ mouse}{a \text{ is } large_m}\ intA_2$$

Hence, subsective adjectives allow us to infer from 'a is $Adj\ \ Noun$' that the underlying object of predication a possesses both its constituents separately: the noun $Noun$, i.e., 'a is $Noun$', as well as the adjective Adj with the proviso it was relativized w.r.t $Noun$., i.e., 'a is Adj_N'. Thus, e.g., 'a is $large_m$' from the example above can be read as 'a is something large assuming mouse-largeness scale'. Analogously to intersective adjectives, we can see that the compound containing a subsective adjective and a noun behaves similarly to the logical connective conjunction.

In NDS, the rules for subsective adjectives are as follows (we omit the generalized elimination rules; Francez 2017):

$$\frac{\Gamma \vdash \mathbf{j} \text{ isa } X \quad \Gamma \vdash \mathbf{j} \text{ is } A_X}{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}\ (subAI) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}{\Gamma \vdash \mathbf{j} \text{ isa } X}\ (subAG\hat{E}_1) \qquad \frac{\Gamma \vdash \mathbf{j} \text{ isa } A\ X}{\Gamma \vdash \mathbf{j} \text{ isa } A_X}\ (subAG\hat{E}_2)$$

The crucial part for the translaton is the premise $\Gamma \vdash \mathbf{j}$ isa A_X in $(subAI)$ rule, which captures the fact that $\mathbf{j}$ is A only under the assumption that $\mathbf{j}$ isa X. Specifically, A_X is a family of adjectives over the set of nouns X. Hence, in effect, it makes the meaning of A dependent on the meaning of X. In other words, the meaning of A is restricted only to a certain class of nouns X. So we will have a different types of largeness: e.g., large human, large insect, large mammal, etc.

As Francez describes it:

> The unfolding of the adjectives can be seen as a purely formal devise to parameterize a subsective adjective: A_X is just a family of adjectives originating from A and parameterized by nouns X. The entailments in (4.14) are the basis for the revised I/E-rules for subsective adjectives. [...] The explicit parameterization replaces the dependency in the original rules. (Francez 2017, pp. 12–13)

In MTT, we can capture this dependency by making the whole type of subsective adjectives range over the X, i.e., (common) nouns. Thus its type will be $\forall\alpha : X.(\alpha \to Prop)$ (see Chatzikyriakidis and Luo 2013, Chatzikyriakidis and Luo 2017a). After the translation, we get:

$$\frac{[\![\Gamma]\!] \vdash j : [\![X]\!] \quad [\![\Gamma]\!] \vdash j : [\![A_X]\!]}{[\![\Gamma]\!] \vdash j : [\![A\ X]\!]}\ (subAI)' \qquad \frac{[\![\Gamma]\!] \vdash j : [\![A\ X]\!]}{[\![\Gamma]\!] \vdash j : [\![X]\!]}\ (subAG\hat{E}_1)' \qquad \frac{[\![\Gamma]\!] \vdash j : [\![A\ X]\!]}{[\![\Gamma]\!] \vdash j : [\![A_X]\!]}\ (subAG\hat{E}_2)'$$

Comments. Rules are justified by the corresponding Σ type rules.

Examples. The following NDS derivation (see Francez 2017, p. 12):

$$\frac{\mathbf{j} \text{ isa elephant} \qquad \mathbf{j} \text{ is small}_{\text{elephant}}}{\mathbf{j} \text{ isa small elephant}}$$

becomes:

$$\frac{\Gamma \vdash j : Elephant \qquad \Gamma \vdash k : Small_E(j)}{\Gamma \vdash (j,k) : (\Sigma j : Elephant)Small_E(j)}\ (subAI)'$$

where $Small_E$ denotes the fact that we use 'elephant-smallness' of type: $Elephant \to Prop$. Note that we cannot derive that there is something small (corresponding to $snd(l) : Small(fst(l))$), only that we have something small w.r.t. an elephant scale.

Acknowledgements. Work on this paper was supported by grant nr. 19-12420S from the Czech Science Foundation, GA ČR.

References

Chatzikyriakidis, S. and Z. Luo (2013). Adjectives in a Modern Type-Theoretical Setting. pp. 159–174. Springer, Berlin, Heidelberg.

Chatzikyriakidis, S. and Z. Luo (2017a). Adjectival and Adverbial Modification: The View from Modern Type Theories. *Journal of Logic, Language and Information 26*(1), 45–88.

Chatzikyriakidis, S. and Z. Luo (2017b). *Modern Perspectives in Type-Theoretical Semantics*. Springer Publishing Company, Incorporated.

Francez, N. (2015). *Proof-theoretic Semantics*. College Publications.

Francez, N. (2017). A Proof-Theoretic Semantics for Adjectival Modification. *Journal of Logic, Language and Information 26*(1), 21–43.

Francez, N. and R. Dyckhoff (2010). Proof-Theoretic Semantics for a Natural Language Fragment. *Linguistics and Philosophy 33*(6), 447–477.

Luo, Z. (2012). Formal semantics in modern type theories with coercive subtyping. *Linguistics and Philosophy 35*(6), 491–513.

Luo, Z. (2014). Formal Semantics in Modern Type Theories: Is It Model-Theoretic, Proof-Theoretic, or Both? In N. Asher and S. Soloviev (Eds.), *Logical Aspects of Computational Linguistics: 8th International Conference, LACL 2014, Toulouse, France, June 18-20, 2014. Proceedings*, Berlin, Heidelberg, pp. 177–188. Springer Berlin Heidelberg.

Martin-Löf, P. (1984). *Intuitionistic type theory*. Studies in proof theory. Bibliopolis.

Negri, S., J. von Plato, and A. Ranta (2001). *Structural Proof Theory*. Cambridge University Press.

Nordström, B., K. Petersson, and J. M. Smith (1990). *Programming in Martin-Löf's type theory: an introduction*. International series of monographs on computer science. Clarendon Press.

Ranta, A. (1994). *Type-theoretical Grammar*. Indices. Clarendon Press.