The Language Multiverse

Marcus Kracht
Fakultät für Linguistik und Literaturwissenschaft
Universität Bielefeld
Postfach 10 01 31
33501 Bielefeld
marcus.kracht@uni-bielefeld.de

1 Introduction

It is obvious that people speak different languages. Today there are (still) some 5000 languages spoken, not to mention the dialects thereof. However, the implications especially for linguistic theory are not really clear to this day. The analysis has always tried to eliminate this diversity in favour of some ideal language ([3]). Especially logical theory has advanced the idea that there is some privileged language from which all others are derived. (I am ignoring the idea prevalent among mathematicians that there is *no* language worth that name and that we are working so to speak *without syntax*.) This has never worked—not even in logic. There is no reason why it should; the evidence speaks against that. Perhaps the resistence to accept a plurality of language stems from a fear that the situation becomes otherwise unmanageable. Actually, as we shall see below, there are some benefits to dealing with multitudes of languages.

The present paper argues that all attempts at avoiding or eliminating the multitude of languages are doomed to fail and that we should instead make the best of it. The author, considering himself both a logician and a linguist, confesses that it may belabour points that are trivial to a logician because they concern the form and not the content of expressions though I sense agreement in the recent [1] that the matter is worth the attention. As logic strives to be a foundational discipline it must also be able to provide answers to (seemingly) naive questions. Needless to say, I do not think the questions are as innocent as they appear.

We shall look first at the scale of the problem in linguistic terms. Then we shall make a turn and look at a somewhat underrepresented aspect: logical laguages. Here the somewhat surprising result is that the same multitude besets everyday logical theory, and that, again there is no other way but to accept the predicament.

I wish to thank the participants of Logic Today 2016, Kai Wehmeier and Maciej Kleczek for useful discussions.

2 How many languages do we speak?

For quite some time now, linguists have studied language learning, multilingualism and code switching. The globalisation and the growing numbers of migrants have lead to growing numbers of communities in which several

languages are being spoken and communication must be established between groups of people of different background. What is not so well known is that even a single language is not homogeneous, and that speakers never speak just one language but rather several of them. However, linguistically there is a resistance to call some of them *languages*.

Technical vocabulary is one instance where a language is being created that looks like a natural language but isn't considered as one. Mathematics borrows lots of words from its host language like "group", "ring", "field", and so on, but gives them completely different meanings. To make matters worse, many words have several meanings in different subfields, such as "normal". Depending on what you talk about, this word can mean vastly different things.

A mathematician will not claim that these words mean altogether different things to them than to other people. They will *not* claim that the word "group" only has the meaning that they have given it. They will say that it has that other meaning *only in special discourse*. Other than that it has the meaning that English speakers decide it should have. So we have factually two (or even more) languages in parallel. The same applies to many other discourses, such as physics, law, sports, and so on.

Standardly, all these variations are included in a dictionary of the language. It is then said that if a word is given a new meaning that given word has many different meanings depending on context. In this way the terminological neologisms are incorporated into the language from which they are derived. This is justified on the grounds that the words keep their morphology. On the other hand technically this underestimates the complexity of the situation. In sciences, the concepts to which the words are linked are translingual. When astronomers decided to revise the meaning of the English word "planet" then this affected virtually all languages at the same time, as German astronomers are now required to use the word "Planet" in the same way, as are Hungarian astronomers with respect to the word "bolygó". Hence, the scientific use of that word is markedly different from its ordinary use, since it does *not* belong to the English language alone!

Linguists have also drawn attention to factors determining the way we speak. Regional variation is referred to as *dialectal variation*, different social relationships give rise to different *sociolects*, or *registers*, and so on. Finally, people differ from each other by slight differences in uses of this or that word; we say they have different *idiolects*.

With all this variety of language we should be asking what the theory

can offer. Unfortunately, language theory largely insists that a language community is homogeneous. While syntacticians have recently given in to the thought that syntax may not be so uniform (a phenomenon that in generative grammar runs under the name of *microvariation*), formal semantics has so far not responded to the challenge. The papers [9, 10] are just a modest beginning.

3 Multilingualism in computer science

The situation in computer science needs special attention because there we actually have a similar situation as in logic but the need for solutions is rather urgent, which is why the matters has been adressed many times. To start, there are probably hundreds of programming languages in this world, each having different incarnations, called versions, which are not always interoperable. It is true that programmers strive for what is called downward compatibility, which means that newer versions still handle code for older versions in the same way, but this is not always possible or practical.

The problem is that work done in one programming language may not be usable across other languages. So one might have to go through the effort again in a new language. This means that a lot effort is wasted. Thus there have been a number of attempts at analysing and solving the problem. To begin with, the notion of an "institution" has been introduced by Joseph Goguen ([6], see also the survey [2]) to analyse the problem. Then there is "Common Logic" (ISO 24707), and initiatives such as Mmt (*meta-meta-theory*, [11]). However, in the present cases unification often, is on the propositional side only; this is especially true for institutions. We translate preserving judgements, not preserving denotation in general. We gain in flexibility, but we lose explanation.

For a linguist it appears futile to try and rise above everybody else. "Meta"ing your way out is not the answer. It just shifts the point of attention one level higher. This much we know from the many failed attempts to provide a logically pure language (see also [3]).

Indeed, there are also projects that do not try to unify the languages but rather allow to integrate them by providing controlled forms of code switching, or to use a more familiar term, by allowing to import code from another language. Such is the case with PHP or JavaScript (in HTML), or the integration of OCaml and CDuce into Ocamlduce (see `http://www.cduce.org/ocaml_manual.html`).

More ambitious projects exist that integrate diverse programming languages.

Seen from the standpoint of the descriptive linguist, we witness side by side attempts at unifying and/or standardising languages (equivalent to prescribing, say, the use of English in documents) and at providing translation tools.

4 Logical pluralism

The problem of linguistic variation may seem far removed from logic. But it is not. Logic is about judgement, and judgements are expressed in a language. Thus there can be no logic in the proper sense of the word without there being a language to begin with. This will raise questions about the relationship between logic and language. Let's start with a simple question, though.

There is, namely, a question lurking in the background. And it is this:

> Is there one logic or many?

For if logic is language bound as its judgments are necessarily expressed in some language then there is a dependency on the language in question. Logicians have sought to circumvent the problem by choosing a vantage point ("*the* language of propositional logic") and then translating everything into that. But the vantage point itself is far from being undisputed. So what happens if we choose another one? If there is no ideal language, no ideal vantage point from which to study the matter, could there not be several logics simply because there are different languages? I guess that point is being made, at least implicitly, by some authors. They claim that the fact that we speak different languages gives rise to different logics. Hartrey Field finds this uninteresting qua logical problem, and I agree. Somehow the differences seem to be just about the syntax and the chosen meanings for the primitives. What he asks for is what he calls an *all purpose logic*. A logic you use when none is suggested by the circumstances. A logic that you would use precisely when you realise that logic depends on language, and you would like your judgements to be about what there is rather than what we call it.

5 The language multiverse

Now let us enter the technical discussion. Let us fix some technical vocabulary. We start with a set E of expressions, and a set M of meanings. They can be as large as we like, preferrably (at least) countably infinite. E may for example contain all finite sequences of Unicode symbols. A *sign* is a pair (e, m), where e is in E and m is in M. A *language* is a subset of $E \times M$. Given $L, L' \subseteq E \times M$, a *translation* is a map $\tau : E \hookrightarrow E$ such that $(e, m) \in L$ iff $(\tau(e), m) \in L'$. (Here, the symbol $\hookrightarrow$ denotes partial functions.)

Definition 1 *The language multiverse over $E \times M$ is the category of all languages (as objects) and translations (as arrows).*

The use of categories serves no theoretical purpose other than providing a succinct statement of the kind of structure we are dealing with. There is an obvious dependency on the sets E and M; it is a dependency that cannot be removed but seems to be rather irrelevant. I shall not comment on it any further. Also, we do not have much to say about the role of M. It will become apparent below that we only need very little structure on M. Some fraction of a typed universe will be enough.

Now let's turn to *grammars*. Grammars spell out the structure of a sign. They are finite devices to generate a language. Let F be a finite set and $\Omega : F \to \omega$. We call Ω a *signature*. A *grammar* is a pair (Ω, I), where I assigns to every $f \in F$ a partial (!) function on $E \times M$ with arity $\Omega(f)$. I can be extended to the terms over the signature. It becomes a partial homomorphism from the algebra of all Ω-terms into a suitable algebra over the signs. Namely, for each $f \in F$, we introduce an $\Omega(f)$-ary function $I(f)$ on the set of signs. $E \times M$ equipped with these functions is a partial Ω-algebra. The grammar can thus be said to impose an algebraic structure on the set of signs.

The set of (Ω-)terms is defined inductively as follows. If $f \in F$ and $t(i)$, $i < \Omega(f)$, are terms so is $ft(0)t(1)...t(\Omega(f)-1)$. Note that terms are written in Polish Notation. Note the special case $\Omega(f) = 0$. In that case f alone is an Ω-term. Terms are evaluated in $E \times M$ using the interpretation function I:

$$(1) \quad *(ft(0)t(1)...t(\Omega(f)-1)) \;:=\; I(f)(*(t(0)), *(t(1)), ..., *(t(\Omega(f)-1)))$$

(The evaluation function $*$ obviously depends on the grammar in question, but we will not explicitly call attention to that fact through notation.) Not

all terms evaluate to a sign. Those that do are called definite. In other words, t is called *definite* if $*(t)$ is defined. Then $L(G)$ is the set of all $*(t)$ where t is a definite Ω-term.

G is *independent* if $I(f)$ can be decomposed into a pair of partial functions of arity $\Omega(f)$, denoted $\varepsilon(f)$ and $\mu(f)$, respectively, where $\varepsilon(f)$ operates on E and $\mu(f)$ operates on M. In that case we have for $\Omega(f) = 2$:

$$(2) \quad I(f)((e,m),(e',m')) \;=\; (\varepsilon(f)(e,e'),\mu(f)(m,m'))$$

where the left hand side is defined if and only if the right hand side is defined.

We give an example. Let F be the set $\{f,g,h,k\}$. The signature is $\Omega : f \mapsto 0, g \mapsto 0, h \mapsto 1, k \mapsto 2$. Write "$\cdot$" for sequence concatenation. Now interpret the function symbols as follows.

- $I(f)() \;:=\; (\mathtt{f},0)$;
- $I(g)() \;:=\; (\mathtt{t,1})$;
- $I(h)((e,m)) \;:=\; (\mathtt{(}\cdot \sim \cdot e \cdot \mathtt{)}, 1-m)$;
- $I(k)((e,m),(e',m')) \;:=\; (\mathtt{(}\cdot e \cdot \wedge \cdot e' \cdot \mathtt{)}, m \cap m')$.

This defines the grammar G. G is independent. This is obvious from the description of the functions. The term $khfg$ can now be evaluated as follows.

$$(3) \quad *(khfg) \;=\; (\mathtt{((} \sim \mathtt{f)} \wedge \mathtt{t)}, 1)$$

The signature is not fixed. This may be technically awkward. There is a fix for that. We introduce the *universal signature* $\mho$, consisting of a consisting of a countable family $F(i)$, $i \in \omega$, of countable sets of function symbols. For each i, $F(i)$ contains the function symbols of arity i. A grammar can be seen as a $\mho$-grammar where almost all $I(f)$ are empty.

6 Structural translations

We have introduced above the notion of a translation. These are meaning preserving maps between languages. Evidently, maps between grammars should induce translations of the generated languages. However, we want that the translations be structural in the sense of the grammars as well. In

general a *structural translation* is a function ζ that assigns to each function symbol f an $\mathfrak{V}$-term function $\zeta(f)$ of identical arity such that

$$(4) \quad I(f)(\sigma(0), ..., \sigma(\mathfrak{V}(f) - 1)) \;=\; \zeta(f)(\sigma(0), ..., \sigma(\mathfrak{V}(f) - 1))$$

where the right hand side is defined if the left hand side is defined, though not necessarily conversely.

While the term function $\zeta(f)$ operates on $E \times M$, we may also think of it as a term. In the sequel we shall not distinguish the two in notation. The translation can be extended to a homomorphism $h\zeta$ of the term algebras. The condition (4) says that if the term t is definite, so is $h\zeta(t)$. If both are defined, then we must have $*(t) = (e, m)$ and $*(h\zeta(t)) = (e', m)$ for some $e, e' \in E$ and $m \in M$. This follows from the requirement that ζ induces a translation of the languages.

We give some examples. Let F be the set $\{f, g, h, k\}$. The signature is $\Omega : f \mapsto 0, g \mapsto 0, h \mapsto 1, k \mapsto 2$. Interpret the function symbols as follows.

- $I(f)() \;:=\; (\mathtt{f}, 0)$;
- $I(g)() \;:=\; (\mathtt{t,1})$;
- $I(h)((e, m)) \;:=\; (\sim \cdot e, 1 - m)$;
- $I(k)((e, m), (e', m')) \;:=\; (\wedge \cdot e \cdot e', m \cap m')$.

Call this grammar G'. It has the same signature as G. G' presents the familiar Polish Notation. Define ζ to be the identity. Then if t is a term and $*(t) = (e, m)$, its translation is $*(h\zeta(t)) = (e', m)$, where e' is e written in Polish Notation.

A second example. Let F again be the set $\{f, g, h, k\}$. The signature is $\Omega : f \mapsto 0, g \mapsto 0, h \mapsto 1, k \mapsto 2$. Interpret the function symbols as follows.

- $I(f)() \;:=\; (\mathtt{f}, 0)$;
- $I(g)() \;:=\; (\mathtt{t,1})$;
- $I(h)((e, m)) \;:=\; (\mathtt{(}\cdot \sim \cdot e \cdot \mathtt{)}, 1 - m)$;
- $I(k)((e, m), (e', m')) \;:=\; (\mathtt{(} \cdot e \cdot \wedge \cdot e' \cdot \mathtt{)}, m \cup m')$.

This defines the grammar V. The translation η works as follows.

1. $\eta(f)() \;:=\; I(f)()$.
2. $\eta(g)() \;:=\; I(g)()$.
3. $\eta(h)(\sigma) \;:=\; I(h)(\sigma)$.
4. $\eta(k))(\sigma, \sigma') \;:=\; I(h)(I(k)(I(h)(\sigma), I(h)(\sigma')))$.

(The notational clutter can be removed if we allow ourselves to write, for example, $\eta(k)(\sigma, \sigma') \;:=\; h(k(h(\sigma), h(\sigma')))$, skipping the interpretation.) It is readily checked that this is a structural translation. This follows from de Morgan's Law that $p \cup q = -((-p) \cap (-q))$.

Definition 2 *The grammar multiverse over $E \times M$ is a category with $\mathfrak{V}$-grammars as objects and structural translations as arrows.*

To illustrate the idea behind this definition, consider writing a text on classical logic. You first want to decide on the set of connectives. Maybe you have a preferred choice, maybe not. Maybe you write: Choose "enough" connectives—but what does that mean exactly? Next you need to decide on the shape of variables; and finally, you need to decide how you write formulae. Do you drop brackets, and if so, where? Do you write binary connectives in infix?

We want to maintain, though, that all that is innocent. We can choose whatever we want. For this to be a reasonable assumption, though, it needs to be assumed that we are actually speaking several dialects of propositional logic but that we have no difficulty to translate between them. And that there is no single "correct" language.

7 Logic as abstraction

We have said above that logic depends on language. If there are several languages it might well be that we also have different logics. Let's see where this leads us. Consider the following rule.

(5) $\quad P \wedge Q \;/\; P$

This rule is expressed in a certain language. Hence it is particular to that language. The letters P, Q stand for propositions. This standard formulation does not mention any grammar, so we must fill in some details. Given

grammar G above we can formulate the rule as follows.

(6) $\mathtt{(} \cdot e \cdot \wedge \cdot e' \cdot \mathtt{)} \;/\; e$

In this form it is about *expressions*, not signs. This rule is called valid if (and only if) for all e and e': if $(\mathtt{(} \cdot e \cdot \wedge \cdot e' \cdot \mathtt{)}, 1) \in L(G)$ then $(e, 1) \in L(G)$. I stress here that validity is thus defined using the grammar of the language and the particular role of the element denoted by "1".

A side remark is in order. (6) is called *admissible* if for every substitution σ, if $\sigma(P)$ is a theorem (= true under all assignments), so is $\sigma(Q)$. This is just to say that validity is defined here as truth preservation, but this is strictly speaking just a convention.

Finally, we may take advantage of the fact that the expression $\mathtt{(} \cdot e \cdot \wedge \cdot e' \cdot \mathtt{)}$ is nothing but $\epsilon(k)(e, e')$. Thus we may once again reformulate the rule as follows.

(7) $\epsilon(k)(e, e') \;/\; e$

In this formulation the dependency on the actual string is removed and the rule can now be abstracted from the grammar. For example, translating it into Polish Notation becomes straightforward. (We are benefitting here from the fact that the rule is actually skeletal.)

In which sense is this rule dependent on $L(G)$, or G for that matter? And furthermore what is the meaning of that dependency if it exists?

As it turns out, rules are heavily dependent on the language and its grammar. The reasons are not obvious at first sight. Firstly, the rule is valid only if the interpretation of the symbols is kept fixed. If $\wedge$ is actually interpreted as $\cup$, the rule is obviously not valid. This is why I have insisted that translations be meaning preserving. However, this solves the problem only half way. We still have the possibility to interpret $\wedge$ as $\cup$. Recall the definition of the grammar V. Translate $k(t, t')$ by $h(k(h(t), h(t')))$. This is meaning preserving. The rule (6) must now be rendered

(8) $\epsilon(h)(\epsilon(k)(\epsilon(h)(e), \epsilon(h)(e'))) \;/\; e$

or in terms of concrete strings:

(9) $\mathtt{(} \sim \mathtt{((} \sim e\mathtt{)} \wedge \mathtt{(} \sim e'\mathtt{)))} \;/\; e$

More is to come, though. Evidently, we cannot base our formulation of the rule on the concrete strings, for they depend on the choice of formation

functions (infix notation vs Polish Notation). Nor can we base them on the terms since the symbols may receive different interpretations.

Finally, notice that translations can also extend the meaning of symbols. If therefore we were to base the rule on the concrete strings, the following may happen. Suppose we extend G to the grammar G^*, which has an additional symbol f' with arity 0 such that $I(f')() = (\mathtt{t}, 0)$. The new language is ambiguous because $\mathtt{t}$ can denote either 1 or 0. What should now be the status of the rule (6)? Recall that we can interpret $\mathtt{t}$ in two different ways: it is the exponent of f (and therefore true) or the exponent of f' (and therefore false). How do we read the assumption $\mathtt{(} \cdot e \cdot \wedge \cdot e' \cdot \mathtt{)}$? Should we say it is true because it is true in at least one instance? Or should we say it is false because it is false in at least one instance? Likewise for the conclusion. And how about the two instances of e in the rule. If we decide to read the first as the exponent of f, are we then committed to the same reading in the conclusion?

Any of the choices lead to different results. For example, consider the rule

(10) $e \;/\; \epsilon(k)(e, e')$

It is valid if we decide to treat a formula as false in case one instance is false (and true otherwise). For either e is a conjunction of one or more $\mathtt{f}$s, then it is false; or it contains at least one $\mathtt{t}$, in which case it has one false instance. This carries over to the conclusion. If we decide that a formula is true if it has a true instance, the rule is however invalid. Take $e = \mathtt{t}$ and $e' = \mathtt{f}$.

Additionally, there is a choice according to the different instances of an expression. I suggest to treat the two occurrences of e as coming from the same term. (In language this is typically the norm; when ambiguous words are being used in an argument then the meaning should be held constant.) However, that in itself suggests that the rule is about terms not expressions. Since logicians generally avoid ambiguity, this difference is immaterial. In general, however, we must use terms. This leads to unambiguous formulations but does not remove the dependency on the grammar, as we have just seen. Alternatively, we can use a regimented language that is uniquely readable, so the map from terms to expressions is injective.

The validity of the rule depended on the meanings that the expressions have. Again, two readings of this are possible. The first is the substitutional reading. Here we read (7) as saying that whatever terms t, t' exist, if $*(k(t, t')) = (e, 1)$ for some e, then $*(t) = (e', m)$ for some e'. To

rephrase this call a term t *true* if $*(t) = (e, 1)$ for some e. Then the rule says that for any two terms t and t', if $k(t, t')$ is true so is t. The other reading is the following. Suppose there are signs (e, m), (e', m') such that $I(k)((e, m), (e', m')) = (e'', 1)$. Then $m = 1$. In the second formulation we quantify over all signs, also those that cannot be expressed using a term.

Again, if the rule expresses a regularity of the grammar then it expresses a regularity of its terms, and so the substitutional interpretation is the only one available.

One thing that I have no yet discussed (as it will be discussed in the next section) is the role of variables. Notice that none of the grammars have genuine variables. And that provides another point of diversion. As matters stand now, variables are proxy for expressions (substitutional interpretation) or signs (nonsubstitutional interpretation). Now if we introduce variables into the language itself, then variables can also figure in the formulation of the rules, which introduces its own problems of ambiguity, as discussed in [5].

The conclusion is the following. There is an interpretive rule (held constant) that regulates what it takes for a rule to be valid in a language or a grammar. Translations can be faithful but may still render rules invalid. This is because they interpret the rule in a larger language that may exhibit counterexamples. There are many examples of that kind. We may for example venture into three-valued logic by introducing a constant that has the value u different from 0 and 1, and subsequently expand the interpretation of our symbols to accommodate for the new values. All this is perfectly sound. But it casts a shadow on our understanding of logic: are we not heading towards logical pluralism that way? I think not. The reason is that if language is no longer universal, so is the logic, since it depends on the language. It gives way to a reasoning beyond language. That in turn is based on logical thinking (dare I say so?). What is its logic? So far it is classical, as far as I am concerned. Indeed, classical logic serves as a last resort, and anything that we do in this essay can be understood using it.

But then where is this classical logic if not on the metalevel? Haven't we just decided that it is used in absence of a language? My idea is that that is not necessarily so. Instead, I argue that classical logic is not on the metalevel, but simply encoded in any two valued logic subject to conditions of unique readability and expressivity constraints (namely that every function on the set $\{0, 1\}$ must be a term function of the induced algebra). To interpret a nonclassical logic we need to come up with a nonstandard translation.

8 Predicate Logic or: What to do with variables

The theory of multiverses can actually be put to use in predicate logic. There is a long literature on the meaning of variables. The questions that appear are: What is the meaning of a variable? And how do we explicate the meaning of a quantifier? The answer that will be given here is not orthodox but derives from the mathematical practice rather than the textbooks on logic.

Consider the following theorem and proof.

THEOREM. There are infinitely many prime numbers.

Proof. Suppose not. Let $\{p(1), p(2), ..., p(n)\}$ be all the prime numbers. Let q be $1 + p(1) \times p(2) \times ... \times p(n)$. Then either q is prime (and is distinct from all the $p(i)$), or it has a prime divisor, r. Then r is distinct from all the $p(i)$. QED

This proof begins by supposing that we have finitely many primes, and shows them to us in the form of a bunch of symbols, n, $p(1)$, $p(2)$, and so on. These symbols *are new*. We have not agreed beforehand that variables look this way. The symbols are arbitrary, and it is clear from the context they serve as variables. On the basis of these symbols, a new symbol, q, is introduced and given a value. It is argued that it is either prime (and then larger than the given primes) or contains a prime divisor r, which must be different from the previously given numbers.

Finally, what is not said but implied is this: whatever concrete value for n we choose, whatever values we then choose for $p(1)$, $p(2)$ up to $p(n)$, the argument remains valid. It does not depend on the actual values chosen for the variables. So in effect, *we are quantifying over language expansions*. Also left implicit is the kind of values we may choose (natural numbers). Thus, not *any* kind of language extension will be considered, but only an extension that is admitted by the context (whatever that means in concrete detail).

So it appears that at any moment we can extend the language by any number of signs. The extension can be definite (q is computed from existing numbers) or indefinite. In the latter case we pick a new symbol and give it an "arbitrary" value. This looks like an incarnation of Kit Fine's arbitrary objects [4]; however the values chosen are real values. They are not of a different kind.

Let's extract the core of this method. Standardly, the meaning of a quantifier is defined in terms of substitution of objects for occurrences of variables.

> $(\forall x)\varphi$ is true in M if for all a, $[a/x]\varphi$ is true in M.

Here, $[a/x]$ denotes the substitution of a for all free occurrences of x. However, φ is a syntactic object, we can't simply put objects there. To make this well-defined Tarski therefore introduced valuations and replaced the clause above by a quantification over valuations:

> $(\forall x)\varphi$ is true in M under a given valuation β if for all $a \in M$, φ is true in M under $[x/a]\beta$, which is β changed at x to give the value a.

A formula is thus evaluated in a structure *together with a valuation*. If the valuation is not part of the structure, there is a problem with the meaning of a variable. While the notation suggests that the variable stands in for an object, in standard logic it must clearly be construed as denoting a set of valuations. [5], in another dissenting analysis, insists that the meaning of a variable is the set of its values. Any of these interpretations raise their own problems.

But suppose there are no valuations and no objects to be put in place of x! And that what happens instead is that we extend the language by a new symbol, or rather, a new sign, like this.

> $(\forall x)\varphi$ is true in M if for all a, on extending the language with a sign (c, a), $[c/x]\varphi$ is true in M.

Here, $[c/x]\varphi$ is the result of replacing the free occurrences of x by c. In this interpretation, variables are metagrammatic symbols. Notice that some details of the extension are not specified. They hardly need to be. In actual fact, the language of predicate logic never was *one* language but a spectrum, depending on function and relation symbols plus their arities. Thus, textbooks actually declare it a language that extends the bare logical language by any functional and relational signature. So the extendability is already built in. We know what to do when we are asked to add a constant. Notice that we have to chose not only a value but also a name for the constant. But even that is immaterial.

Now what happens to the symbols once the proof ends? They get removed. In that way they can be used anew. Thus we need to consider what happens if the symbols get removed. If a name is removed, so is the sign and hence the value that is assigned to the variable. The scope of the quantifier is determined by the moment where the symbol that it introduces is removed.

Curiously enough, the analysis presented here is more or less the one suggested by Frege (see [12] for a discussion) and can also be found in [7]. For these authors, there were no open formulae. Hence $\exists x.\varphi$ could not be formed from φ. Instead, a quantifier was introduced by removing instances of a constant (!) and putting variables in its place. The difference between Frege's approach and the proposal made here is that we take the language as basically not specified in advance; and that meanings are specified by quantifying over language extensions. This makes the formulation of compositionality awkward, since compositionality is defined within a single language. Wehmeier suggests in [12] to quantify over model structures which differ from each other in the values of constants. This allows to keep the language constant.

There is a perfect parallel with functional programming. In functional programming, a variable is declared once and its value may not be altered. See XSLT for a perfect illustration of this [8]. In this language, there is no distinction between constants and variables, everyhing is called variable (or parameter, which is a variable whose value is given from outside). "x := x+1" is meaningless. Reassignment of values can nevertheless be done. Here is an example to get the result of $x := x + 1$.

1. Create a new variable j with value $x + 1$.
2. Remove the name x.
3. Create a new (!) variable x with value j.
4. Remove the name j.

Notice however that one cannot do this in XSLT, because scopes have to be nested.

9 Conclusion

The present paper argues that there never is just one language but rather a whole multitude of them, and that the way to handle this multitude is

not by creating a unique perfect language, or metalanguage, but rather by providing ways to translate between them. Languages thus form a category under the translation maps.

References

[1] Francesco Belluci and Ahti-Veikko Pietarinen. Existential Graphs as an Instrument of Logical Analysis. *The Review of Symbolic Logic*, 9:209 – 238, 2016.

[2] Răzvan Diaconescu. Three decades of institution theory. In Jean-Yves Béziau, editor, *Universal Logic*, pages 309–322. Birkhäuser, 2012.

[3] Umberto Eco. *The Search for the Perfect Language (The Making of Europe)*. Wiley-Blackwell, 1997.

[4] Kit Fine. Arbitrary objects and natural deduction. *Journal of Philosophical Logic*, 14, 1985.

[5] Kit Fine. *Semantic relationism*. Blackwell, London, 2007.

[6] Joseph Goguen and Rod Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the ACM*, 39:95–146, 1992.

[7] David Hilbert and Paul Bernays. *Grundlagen der Mathematik I*. Springer, Berlin/New York, 1934.

[8] Michael Kay. *XSLT 2.0 and XPath 2.0. A Programmer's Reference*. Wrox, Indianapolis, 4 edition, 2008.

[9] Marcus Kracht and Udo Klein. The Grammar of Code Switching. *Journal of Logic, Language and Information*, 23:313–329, 2014.

[10] Marcus Kracht and Udo Klein. Notes on disagreement. In Daniel Gutzman, Jan Köpping, and Cécile Meier, editors, *Evaluations – Denotations – Entities. Studies on Context, Content and the Foundation of Semantics*, pages 276–305. John Benjamin's, Amsterdam, 2014.

[11] Florian Rabe. The Future of Logic: Foundation Independence. *Logica Universalis*, 10:1–20, 2015.

[12] Kai F. Wehmeier. The Proper Treatment of Variables in Predicate Logic. 2016.