Co-Representing Structure and Meaning of Mathematical Documents

Michael Kohlhase and Mihnea Iancu
Computer Science, Jacobs University Bremen, Germany
http://kwarc.info/

October 26, 2015

Abstract

The digital revolution for documents in science, technology, engineering, and mathematics has largely been restricted to simplifying document access for human readers, but has not fulfilled the promise of computer-supported knowledge appropriation.

We claim that this is largely a consequence of the lack of explicitly represented document semantics that computers could act upon. Automated semantics-extraction that could remedy this shortcoming still has two bottlenecks: the lack of language processing algorithms and – less commonly recognized – that of comprehensive representation formats that can adequately model the functional structure of documents and that of the knowledge conveyed by them.

In this paper we analyze phenomena of common mathematical language and derive requirements for representation formats for mathematical documents. We show how these requirements can be met by the OMDoc (Open Mathematical Documents) format and compare it to related approaches.

1 Introduction

Mathematical knowledge is rich in content, sophisticated in structure and technical in presentation. Its conservation, dissemination, and utilization constitutes a challenge for the community and an attractive area of inquiry. Moreover, mathematics plays a fundamental role in Science, Technology, and Engineering (STEM). Therefore, the methodology for representing mathematical documents may be applicable, more generally, to STEM documents: Effectively, we view mathematics as a test tube for STEM.

How mathematics will look in the 21^{st} century remains an open question, but we conjecture that it will become a computer-supported activity, whose major components – visualized in the form of a closed loop driving a creativity spiral by Bruno Buchberger in 1995 – are supported by systems that synergize by exchanging representations of mathematical knowledge.

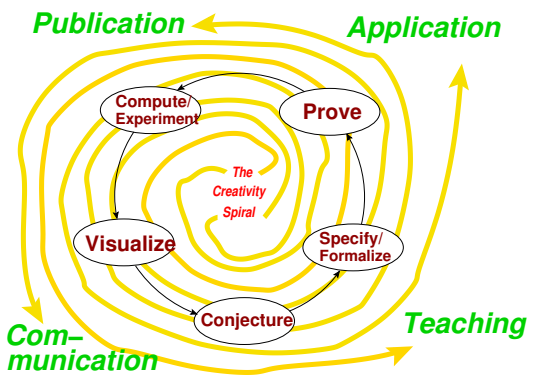

Figure 1: The mathematical Creativity Spiral

Currently, most mathematical knowledge is laid down in the form of documents ranging from research articles over engineering whitepapers to math textbooks. Even though these are often encoded electronically – as digitizations of printed material, or born-digital in the form of PDF generated from LaTeX (predominantly in math research), but also in other formats (in education and engineering) – they are usually only consumed by human readers. If we want to raise this treasure chest of knowledge and use it to support "doing

mathematics" by computer (see Figure 1), we have to recover the knowledge in these documents in a form that computers can act on. This knowledge-recovery process is essentially the syntactic/semantic analysis phase of natural language processing instantiated to mathematical documents. This analysis process is traditionally viewed as a transformation from utterances/documents into a knowledge representation, traditionally some logical system that allows to model inference processes, e.g. first-order logic.

In this paper we focus, not on the translation process from mathematical natural language into knowledge representations, but on the target format, namely the representations of the resulting knowledge. At the first glance, first-order logic seems a plausible target representation format for mathematical knowledge, after all, it is (together with axiomatic set theory [Ber91]) the generally accepted theoretical foundation of mathematics.

However, it is important to note that the transformation process and target format mutually constrain each other. For instance, if we use first-order logic as a target format, then we cannot (realistically) have a compositional treatment of verb phrases and determiners. Richard Montague solved this problem by extending the target format to a higher-order logic based on λ-calculus [Mon74]. At the inter-sentential level, we see that anaphoric references cannot be treated properly in static logics (like first-order and higher-order logic), which has led to the adoption of dynamic logics in computational linguistics. Stepping up one more level, we need still other representational devices – e.g. discourse relations – to account for text/document/collection coherence (see e.g. [JM09, chapter 21]). Note that the levels of representation interact with each other, for instance it is important to know in which dialogue turn we are to bind references or even disambiguate the word meanings. Therefore, the semantics construction and analysis algorithms profit from integrated knowledge representation across the levels.

But all of the above concentrate on "general natural language", which in practice almost exclusively means non-STEM documents. Our experience with STEM documents and mathematical ones in particular have shown that while they retain most of the linguistic complexities (see also [Bau99, Wol13]), mathematical documents have their very own peculiarities that mean traditional natural language processing tools are not directly applicable (see [Wol12]). On the other hand, mathematical documents are (often) more rigorous and have more explicit representations of references and discourse structure. Therefore it makes sense to develop representation formats that take these into account. These formats can then act as targets for semantic analysis processes and in that guide the development of algorithms; to facilitate this is the aim our article.

Specifically, we derive a set of requirements (Section 3) that a representation format for STEM documents needs to satisfy from a set of phenomena peculiar to (natural) mathematical language (Section 2; building on and extending [Wol13]). In Section 4 we propose the OMDoc format as one way to satisfy (many of) these and relate it to other approaches in Section 5. Section 6 concludes the paper.

2 Phenomena of Mathematical Vernacular

Common Mathematical Language (CML) is the sublanguage of natural language used for expressing mathematical knowledge. Like any sublanguage, it has developed particular linguistic devices that make it more suitable for expressing mathematical thoughts and knowledge. These include grammatical constructs, idioms and linguistic conventions but also structural adaptations with significant effects on the fundamental properties of the language. The integration of formulae as well as the use of symbolism and complex notations make mathematical discourse more concise and readable for experts while at the same time making it seemingly impenetrable for neophytes. Note that CML includes the ability to extend its vocabulary, so once readers know enough CML, they can bootstrap and understand any new piece of CML on the basis of prior knowledge.

From the perspective of natural language understanding, this means that CML raises very particular challenges and as a consequence, standard processing tools designed for general natural language often do badly for math [Wol12]. To understand why that is, one must first analyze the linguistic phenomena that are specific to mathematical discourse.

In this article, we do not want to give a complete account of the particular syntactic and semantic phenomena of CML – we refer the reader to the work reported in [Bau99, Zin04, Wol13, Gan13] – but we want to highlight some issues that pose challenges to the target formats of semantics construction. This issue has not been in the focus of the literature so far as the work concentrated on semantics construction/extraction algorithms and the particular logical systems tailored to them. So let us start with a high-level assessment of CML before we go into particular challenges in the subsections.

From a high level view, mathematical texts discuss mathematical objects and their relations using textual representations[1]. Critical parts of assertions or proofs are often reduced to writing down or manipulating syntactic representations of mathematical concepts. At the same time, mathematical documents are written with humans in mind as the target audience so the harsh formalism of mathematical notions is often softened by a carefully constructed narrative structure, examples, intuitions and high-level overviews.

This means that CML must satisfy two conflicting requirements. Firstly, the intrinsic properties of mathematics require mathematical language to be rigorous and precise. Secondly, human authors and readers aim for concise and intuitive results that are easy to write and understand. In fact in many cases the value of proofs, for instance, is measured by their subjectively perceived beauty rather than their precision or even correctness.

Therefore, in practice, the form of each narrative projection of mathematical results depends on notational, didactic, and linguistic considerations. It is an important observation that mathematical texts can only be understood as a projection of abstract mathematical results. This is of course true for any language as it is just a code for encoding information. However, in mathematics, the background knowledge is crucial because mathematical texts often require a more active reader.

In the interest of conciseness, mathematical discourse may carry that to an extreme. In practice, ambiguities are accepted if they can be resolved from the context. The correct meaning is left to be inferred by the reader. The understanding process often draws on background knowledge of concepts and notations. It can also involve non-trivial inferences for filling in missing proof steps. Moreover, CML uses symbols, concepts and notations introduced somewhere else (sometimes later) in the text. Even for expert human readers, this is a very involved task and, therefore, it raises significant challenges for computers.

In the rest of this section, we will look at some key language phenomena that are specific to mathematical discourse and distinguish it from generic natural language. We will distinguish phenomena at three levels of granularity: representations of mathematical objects at the phrase level, mathematical statements at the discourse level, and context phenomena at the document or collection levels.

2.1 Phrase Structure

P1: Formulae as Linguistic Objects Formulae are an essential part of mathematics and, consequently, are also an important part of mathematical discourse. What is unique about formulae in CML is their deep integration into narrative text to the point where they became integral parts of the language. Formulae often appear interspersed within the text, e.g. in (1). More rarely, text can occur inside formulae, such as in (2), and as we see in the slightly contrived (3), text and formulae can recurse freely.

(1) *For all* $a, b \in \mathbb{N}$ *there exists a number* m *such that* $a|m$ *and* $b|m$.

(2) $\{x \in \mathbb{N} \mid x$ *is prime*$\}$

(3) $\{x \in \mathbb{N} \mid$ *the factors of* x *sum to* $2x\}$

P2: Formulae as Grammatical Objects Linguistically, formulae can take on a variety of roles: Formulae can usually be replaced by their **verbalization**, i.e. a mathematically equivalent textual

[1]For the purposes of this article we neglect the fact that CML also uses tables, plots, charts, and diagrams besides the textual modality and leave their study to future work.

representation, typically achieved by reading out the formula as a sentence. Therefore, formulae routinely act as special phrases in CML and take on the grammatical role of their verbalization[2]. For instance in (4) $A \cup B$ functions as a placeholder for "*the union of* A *and* B" and takes on its linguistic role.

Most commonly, formulae appear as noun phrases, e.g. in (4), but can also be of sentence type (5). More rarely, and somewhat colloquially, formulae can be verb phrases such as $\parallel$ in (6) where $\parallel$ has been introduced as a symbol for "*misses*" (or "*is disjoint with*") earlier. In (7), the existential quantifier $\nexists$ modifies its argument noun phrase both mathematically and verbally. In (8), a complex formula acts as a numeral.

(4) $A \cup B$ *meets* $B \cap C$ *on an open ball.*

(5) f *is continuous and* $f(3) = 9$.

(6) *primes* $\parallel$ *evens* > 2

(7) $\nexists$ *greatest prime number*

(8) *Peter has* $2k + 1$ *apples.*

P3: Notations and Verbalizations Mathematical objects usually have two concrete textual realizations, *notations* and *verbalizations*. While functionally equivalent, the two fundamentally differ in form and usage. Verbalizations, such as "*plus*" or "*the set of natural numbers*" are natural-language based representations of mathematical notions. They are commonly found in mathematical texts but only rarely inside formulae, usually when there is no standardized, nice-looking notation, e.g. in (3). Notations, such as infix $+$ for addition or $\mathbb{N}$ for the set of natural numbers, are associated with the formal representation in mathematical language. They are usually used within formulae although they are also often interspersed throughout the narrative text.

P4: Referential Meaning In natural language, the meaning of lexical entries is grounded in the experience of the interlocutors as embodied agents in the world. Even though it is very difficult to define a concept, e.g. of a chair, in all its aspects, we all share a broad consensus on this matter. In mathematics this is different – possibly because it is difficult to directly experience mathematical objects and concepts. Mathematical concepts and objects are defined in explicit definitional statements (see Section 2.2) from previously introduced concepts or objects or declared by selection from non-empty sets with their properties further refined by assumptions or axioms. The only possible exceptions are simple mathematical concepts like $\mathbb{N}$, or $\mathbb{R}^n$, which can be directly experienced by counting and as space. Forgoing a rehash of philosophy of mathematics[3] we subscribe to a referential theory of meaning, where the meaning of lexical items (mathematical symbolism and technical terms) is given by referring to the statements or phrases that introduce them (we call these declarations; see below); see [WG10, WGK11] for linguistic studies that justify this view.

P5: Declarations: Naming Objects Locally Symbolic names provide a concise textual representation for mathematical objects in CML. Being, in essence, atomic formulae, they can be used both in formulae and in text. Since mathematical practice relies heavily on syntactic manipulations of formulae, naming objects is essential and, in practice, common.

There are various forms of declarations in CML: (9) shows occurrences of binders (here the quantifiers $\forall$, $\exists$) in a (first-order logic) formula. Here the binders merely declare the identifiers ε, δ, x, and y, whose scope ranges over the body of the binders (the part after the .). (10) shows a CML version of (9), where the binders are verbalized and the declarations contain *domain restrictions* of various forms, e.g. $\delta \in \mathbb{R}^+$ and $\varepsilon > 0$. We think of the first as a *type declaration* (aka. sortal restriction in linguistics) and the second as a *propositional restriction*; we distinguish them as they are usually handled differently in reasoning: types are built into the substitution mechanism (type checking, usually kept implicit), whereas propositional restrictions are subject

[2]Note that in mathematical logic the term "formula" is normally used in a more restricted sense than the sense in which it is used here, namely to refer to symbolic expressions that behave like sentence phrases – whereas symbolic expressions that behave like noun phrases are called "terms".

[3]. . . where the jury is still out even on matters like the existence of the natural numbers; see e.g. [BP64]

to the general reasoning process. The declaration "$x, y \in \mathbb{R}$ *with* $|x - y| \leq \delta$" has a compound restriction and is placed after x and y have been used; a rather common practice used to make the syntactic structure of the assertion easier to comprehend. For verbalized binders, the scoping of introduced identifiers is often ambiguous. For instance, in (11) the scope of n extends to the second sentence. The fact that the construction is that of a "donkey sentence" [KR93] suggests a dynamic treatment.

(9) $\forall\varepsilon.\exists\delta.\forall x, y.|x - y| \leq \delta \Rightarrow |f(x) - f(y)| \leq \varepsilon,$

(10) *For all* $\varepsilon > 0$*, there is a* $\delta \in \mathbb{R}^+$*, such that* $|f(x) - f(y)| \leq \varepsilon$ *for all* $x, y \in \mathbb{R}$ *with* $|x - y| \leq \delta$.

(11) *If* $n > 10$ *is prime, then it is odd. Moreover* n *ends in one of the digits* 1, 3, 7, *or* 9.

(12) *The depth* $d(T)$ *of a tree* T *is defined as* ...

(12) contains an example of a nested declaration commonly used for relational nouns (which occur often in CML to verbalize functional objects) and their notations. (12) introduces the verbalization of the "*depth*" of a tree together with the notation $d(\cdot)$. T (the tree) is a parameter to the definition and is named in a similar way: "*a tree* T" gives the tree a name which can be referenced in the notation (and definition) for the depth. Grammatically, the names in declarations are usually appositions to the verbalizations, another example for **P2**.

Note that in all cases the declarations have local scope, i.e. the names are only accessible within the sentence or (more commonly) the containing statement (see Section 2.2). Scoping of global names (introduced by definitions) is subject to the context level (see Section 2.3).

P6: Ambiguity In practice, mathematical notations have implicit and loosely defined scope and precedence which help readers disambiguate formulae. For instance "$\sin x/y - z$" is ambiguous between "$\sin(\frac{x}{y}) - z$", "$\sin(\frac{x}{y-z})$", and "$\frac{\sin(x)}{y-z}$" but, for a reader with the required mathematical background knowledge its meaning is clear (usually because one of the readings is nonsensical mathematically). In other cases, ambiguities can be caused by different operators having similar notations, for instance function application and multiplication. However, $f(a(b-c)) = 3$ would be considered clear by most mathematicians due to well established naming conventions, by which f is preferentially used for functions and a and b are used for individuals. For a more in-depth study of ambiguities we refer the readers to [Wol13, Zin04, Gin11, Gan13] and the literature referenced in these.

P7: Elision Moreover, some operators such as Σ, lim, $\int$ have complex notations where arguments can be omitted and have implicit default values. For instance, $\log n$ is usually taken to be $\log_{10} n$ in most contexts and $\log_2 n$ in others; given a sequence $a_0, a_1, \ldots, a_k$ in the context, Σa_i is automatically understood by an experienced reader as $\Sigma_{i=0}^{k} a_i$.

Additionally, the arity of mathematical notations can be flexible as there are symbols that take sequences as arguments. This includes sets, lists, vectors, matrices but can also be extended to most infix binary operators in the presence of associativity.

2.2 Discourse Structure

P8: Mathematical Statements At the discourse level, CML is more explicit in delineating the role of important paragraphs than common natural language. Definitions, theorems, lemmata, corollaries, examples, or proofs are often marked up – even in informal text – using specific fonts, emphasis or keywords and numbers for referencing. Moreover, statements often have specific relations to others depending on their type. For instance, a proof proves an assertion, a definition defines one or more concepts and a corollary follows a theorem. Furthermore, giving a statement the type of a "*corollary*" implies the proof (from the theorem) is trivial, while a classification as a "*theorem*" implies that a proof exists, whether or not it is given in the text.

P9: Inline Statements Even though statements are usually paragraph-level structures, abbreviated forms also appear at the phrase level. Consider the situation in (13) where we use the existence and uniqueness of a solution to a (differential) equation to define a new concept (the

exponential function) in a parenthetical phrase. This construction is usually considered to be mathematically equivalent to the combination of the explicit statements (14) and (15).

(13) ***Theorem 7.1***: *There is exactly one solution (the* ***exponential function*** $f(x) = e^x$*) to the equation* $f' = f$ *with* $f(0) = 1$.

(14) ***Theorem 7.1***: *The equation* $f' = f$ *has exactly one solution with* $f(0) = 1$.

(15) ***Definition 7.2***: *We call the function* f *with* $f' = f$ *and* $f(0) = 1$ *the* ***exponential function*** *and write* $f(x) = e^x$.

(16) ***Definition 3.17*** *The number* e *is an important mathematical constant, approximately equal to* 2.71828*, that is the base of the natural logarithm. It is the limit of* $(1 + \frac{1}{n})^n$ *as* n *approaches infinity, an expression that arises in the study of compound interest, and can also be calculated as the sum of the infinite series* $e = 1 + \frac{1}{1} + \frac{1}{1\cdot 2} + \frac{1}{1\cdot 2\cdot 3} + \frac{1}{1\cdot 2\cdot 3\cdot 4} + \cdots$

The last example (16) shows that inline statement statements can be intricately nested to form complex, but concise statement-scapes, featuring a (primary) definiens, alternative definitions, properties, and even hints for calculations.

2.3 Context/Document Structure

P10: Object-Oriented Notion of Context Mathematics is a very heterogenous field and mathematical texts in different areas routinely use distinctive notations and vocabularies. This is a natural consequence of the distributed nature of mathematical practice and of mathematics' dynamic vocabulary. In practice, mathematical texts remain understandable due to a complex notion of document context that resembles an object-oriented model. In this analogy, objects correspond to mathematical articles, books, or theories (e.g. theory of algebra) which encapsulate definitions, theorems, notations, etc.

Then, as mathematical documents build on existing knowledge, they implicitly *inherit* concepts, notations and verbalizations from such sources to create a specific mathematical vocabulary. Multiple inheritance is common but, due to its informal nature, conflicts (such as diamond problems) are either left to the reader to resolve or resolved by overriding the ambiguous notation or verbalization for the scope of the paper. Symbol scope is mostly stack-like with later, more local, declarations shadowing earlier ones but this behavior can be refined by explicit or implicit imports, as in example 17.

(17) *We use the notation of [BrHa86], with the exception...*

Context becomes even more complicated when accounting for the dynamic vocabulary inside the paper where newly defined notions may be used later in the text.

Moreover, in mathematical discourse there are *local* contexts that add variables or assumptions and can be referenced only in a limited scope – e.g. in (18). This is a natural linguistic projection of variable binding in formulae and avoids writing awkwardly long sentences by extending the scope over several sentences, paragraphs, or even sections. However, it can lead to ambiguous scoping.

(18) *Suppose* $p_1, p_2, \ldots, p_k$ *are all the primes. Then, let* $P = \Pi_{i=1}^{k} p_i + 1$ *and* p *a prime dividing* P*. But every* p_i *divides* $P - 1$ *so* p *cannot be any of them. Therefore* p *is a new prime. Contradiction, so there are infinitely many primes.*

P11: Framing An important property of mathematical thought is *framing* certain notions in terms of others. In mathematical discourse objects of interest are often viewed in terms of already understood structures and creative use is made of this perspective. This allows relating seemingly distinct mathematical fields as well as establishing new results; see [KK09, Koh14b] for a discussion.

The set of integers can be viewed as a group under addition, subsets of a fixed set as (characteristic) functions and functions as sets (of input/output pairs). More advanced results include interpreting a Boolean algebra as a field of sets via Stone's representation theorem or embedding an abstract manifold into some Euclidean space using Whitney's embedding theorems.

Framing is deeply ingrained in mathematical thought and naturally transpires into the presentation of mathematical results. Consider for instance (19) and in particular the adjective "*discrete*", which applies to metric or topological spaces[4]. But it is well-known that a normed vector space can be viewed as a metric space: $d(x, y) := \|x - y\|$ is a metric. So we can felicitously utter (19) to any interlocutor who can frame a normed vector space as a metric space. (20) is similar, only that the concept of a "*ball*" is a concept in metric spaces and the concept of "*open*" sets in topological spaces – here we have a case of "nested framing" and need the additional framing that metrics induce topologies.

(19) *A normed vector space* $(V, \|\cdot\|)$ *cannot be discrete.*

(20) $|\cdot|$ *is a norm on* $\mathbb{R}$ *. . . Let* $B \subseteq \mathbb{R}$ *be an open ball, . . .*

Mathematical adjectives (and nouns in declarations) are notoriously prone to be applied via frames – see also [Koh14a], so framing appears as a important challenge in the understanding and semantic representation of mathematical texts.

P12: Recaps (via renamings) Standalone mathematical documents (e.g. articles or books) have a standard narrative structure: a title page with the abstract, an introduction followed by the main part that presents the body of knowledge the document is written to convey (the payload of the document), which is followed by conclusions. Often, such documents also contain a presentation of related work and a recapitulation of the conceptual foundations to make the document self-contained and/or introduce the concepts and notations needed to understand the payload – for lack of a better word we call such document fragments **recaps**.

(21) *We will use the term* ***monoid*** *for a set* G *with an associative operation* $\bullet$ *that has a unit element and call it* ***group*** *if* $\bullet$ *admits inverses. Finally, a* ***ring*** *is an additive commutative group whose multiplicative structure is a monoid, such that the two operations distribute.*

This example shows that recaps have a specific, telegraphic style: A large number of concepts may be curtly introduced. Explanations and examples are rare, references and renamings are common. In principle recaps are not designed to be self contained or understood by a novice. Instead they offer a reminder for experts and references which take the place of rigorous descriptions. Therefore, understanding these parts requires the ability to *map* the concise presentation from the recap to the knowledge found in the referenced documents.

For instance, the recap section in [CS09] contains the sentence (22). This is a telegraphic version of the full definition, which is given in the literature. Actually [CS09] continues with an overview of the literature, citing no less than 12 papers, which address the topic of accelerated Turing machines. One of these supposedly contains the formal definition, which involves generalizing Turing machines to timed ones, introducing computational time structures, and singling out accelerating ones, e.g. using (23).

(22) *An* ***accelerated Turing machine*** *(sometimes called* ***Zeno machine****) is a Turing machine that takes* 2^{-n} *units of time (say seconds) to perform its* n^{th} *step; we assume that steps are in some sense identical except for the time taken for their execution.*

(23) ***Definition 1.3****: An* ***accelerated Turing machine*** *is a Turing machine* $M = \langle X, \Gamma, S, s_o, \square, \delta\rangle$ *working with with a computational time structure* $T = \overline{[\{t_i\}_i, <, +]}$ *with* $T \subseteq \mathbb{Q}_+$ *(*$\mathbb{Q}_+$ *is the set of non-negative rationals) such that* $\sum_{i\in\mathbb{N}} t_i < \infty$.

[4] stating that the metric or topology is discrete (i.e. separates points), which a norm cannot do, since it has to scale with scalar multiplication (this is the core of the assertion in (19)).

2.4 Not covered in this Article: Proofs

P13: Proofs are one of the distinguishing features of mathematical documents, they consist of text fragments of various sizes – ranging from single words as in (24) over meta-instructions as in (25) to complexly structured argument structures that can span a whole chapter or even an entire book[5].

(24) ***Proof***: *Trivial* □

(25) ***Proof***: *The proof is left to the reader as an exercise.* □

(26) *. . . $f(n)$, which must obviously be positive.*

(27) *. . . thus we have . . . , hence we have proven the assertion.*

Proofs straddle the three levels we used to structure our phenomena: for instance, the "proofs" in (24) and (25) are at the statement/discourse level (they occupy their own paragraph initiated and terminated by special visual cues). In (26) we see an inline assertion (of positivity of $f(n)$), where the proof is represented by the single word "*obviously*". Usually proofs are structured into proof steps, which are often at the sentence level and which consist of an (intermediary) assertion with a justification. In larger proofs, proof steps can be interspersed by proper statements like definitions, assertions, statement of subgoals, subproofs, etc. The rhetorical relation between all of these are given by the justifications, which can be explicit, lexicalized by special termini like "*thus*" and "*hence*" in (27), or (very often) completely elided. The rhetorical structure mediates complex scoping structures that determine the visibility of identifiers/names and the accessibility of results.

Already this brief tour de force around the phenomenon of proofs shows that they are complex and interesting linguistic/semantic structures, which we cannot fully cover in this article, so we leave a more thorough study to future work.

3 Requirements for a Target Language for Semantics Construction Analysis

In this section, we will develop some additional requirements for semantic target formats. These come from management, efficiency, and interoperability considerations induced by the observation that linguistic studies of mathematical documents is a niche subject, where researchers have to collaborate to be able to create the necessary linguistic resources like gazetteers, semantic lexica, etc. This leads to the following three (distinct, but interrelated) requirements:

R1: Flexiformality We discussed above the dual role of mathematical knowledge representation, to be precise and rigorous – close to the semantics – on the one hand and concise and intuitive – understandable by humans – on the other. Arguably, it is these irreconcilable requirements that give rise to two kinds of representations usually associated with formal and respectively informal mathematics. While these two kinds of representation are meant to refer to the same platonic world (the world of mathematics) it has proven difficult to formally express this connection. One can easily relate a formal document with its informal counterpart or a formal proof with its informal version but, at least intuitively, the relations are more complex and fine-grained than that.

This becomes especially complex when considering partially semanticized documents produced by (math-specific) natural language processing tools. There, only some parts or aspects of the document semantics are explicit, while others parts remain **semantically opaque** – i.e. the meaning is inaccessible for current computational systems. Given the prevalence, in the real world, of informal documents and of efforts to semanticize them [SK08], being able to represent such partially semanticized documents is essential.

[5]The proof of the classification of finite simple groups, which consists of tens of thousands of pages in several hundred journal articles written by about 100 authors, published mostly between 1955 and 2004 is a somewhat extreme example.

The flexiformalist view [Koh13] extends the usual dichotomy of formal and informal documents and emphasizes the importance of both representations and the integration between them. Therefore, a *flexiformal* document merges the human-oriented presentation (narration) of a document with its underlying semantics. There are no particular requirements for how much semantics or narration should be present in a document for it to be considered flexiformal. This means that all documents are considered flexiformal with fully formal and fully informal documents representing the two extreme, degenerated manifestations of flexiformality.

R2: Pluralism Most traditional target formats for semantics construction mix structural aspects with logical ones. For instance, dynamic logics like DRT [KR93] or DPL [GS91] address structural issues like the accessibility of discourse referents with logical ones like the truth functionality of logical connectives. Montague-style [Mon74] systems combine structural issues (using λ-calculi to achieve compositionality of semantics construction) again with certain – often non-standard – assumptions about the propositional level.

Correspondingly, semantics construction algorithms require (or have only been investigated for) special target formats, and thus become non-interoperable. To alleviate the interoperability problem, a target format should be *pluralistic* – i.e. be able to accommodate multiple "logical languages". Pluralism can be achieved in two ways: *i*) by combining all the desired traits of the various semantic formats into a single, extremely expressive language that contains all other formats as sub-languages (we call this **homogeneous pluralism**), or *ii*) by making the logical language a parameter in the representation format and (optionally) relate different logical languages via translations (we call this **heterogeneous pluralism**). The homogenous approach has been prevalent in linguistics, leading to systems like λ-DRT [KKP96] or dynamic Montague grammar[GS90], which can become complex and unwieldy. The heterogeneous approach has been gaining traction in formal methods, where the zoo of logical systems and the ensuing interoperability problems are even greater than in natural language semantics – see [Pfe01, Mos05, CHK+11] for discussions.

R3: Underspecification Under the name "underspecification" the natural language semantics community discusses a set of techniques for dealing with ambiguity by deliberately omitting information from linguistic descriptions to capture several alternative realizations of a linguistic phenomenon in one single representation and how to process such underspecified representations without multiplying them out. Approaches range from lexical techniques for polysemous words [Pus98], packed formula representations [DT00] to meta-languages that describe a set of formulae [KTP10]. Even though mathematical language is often regarded as virtually non-ambiguous, semantic target formats should support underspecified representations. This is particularly important for the lexical level, where polysemy is rampant.

4 OMDoc

OMDoc is a representation format for STEM (Science, Technology, Engineering, and Mathematics) knowledge developed by the authors. A major difficulty in representing STEM documents in general (and mathematical documents in particular) is the tension between the formal intent and the narrative presentation of the documents. To achieve this integration, we have developed the notion of *flexiformal representations* – representations of flexible formality; covering the whole spectrum from informal but rigorous to fully formal in a logical system; see [Koh13] for details.

The base of any representation language is a model of the data to be represented. This model is then implemented in a concrete language that supplies markup primitives that allow to express the data. There are two general paradigms for markup: *embedded markup*, where the data is interspersed with control sequences that express the data model and *standoff markup*, where the objects and relations of the data model are specified by pointing into a document with the raw data. Both paradigms have their particular advantages and can be transformed into each other by standardized tools (metadata harvesters and data aggregators). For many applications mixed forms are

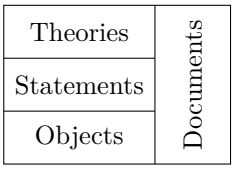

Figure 2: OMDoc Document Model

most suitable.

As OMDoc represents the content and form of STEM documents, the data model of OMDoc is actually a "document model", i.e. the objects are classified text fragments and mathematical knowledge items. The OMDoc document model consists of a narrative part (on the right in Figure 2) and a content part which consists of mathematical knowledge items at three distinct levels: objects (mathematical formulae), statements (definitions, assertions, proofs, etc.), and theories (on the left side of Figure 2). The OMDoc data model provides specific relations between these document and knowledge items that can be used in applications.

Most of the item classifications and relations are directly implemented in the XML-based[6] OMDoc representation language as custom elements and attributes. Only secondary relations (e.g. document metadata relations) have been off-loaded to an extensible metadata mechanism.

In the context of the previous sections, the object, statement and theory levels roughly correspond to the phrase structure, discourse structure, and context level respectively; we now discuss these in detail and show how the CML phenomena identified in Section 2 can be modeled in OMDoc; Figure 3 gives an overview over the situation.

#	Phenonmenon/Requirement	Realization	Sec.
P1	Formulae as Linguistic Objects	content/presentation MathML	4.1.1
P2	Formulae as Grammatical Objects	RDF/RDFa markup	4.1.1
P3	Notations and Verbalizations	notation/verbalization definitions	4.2.1
P4	Referential Meaning	content MathML and CDs/theories	4.1.1
P5	Declarations: Naming Objects Locally	phrase-level markup	4.1.4
P6	Ambiguity	parallel markup	4.1.2
P7	Elision	reconstruct & mark as elided	4.1.3
P8	Mathematical Statements	statement-level (parallel) markup	4.2.2
P9	Inline Statements	parallel markup, reference full version	4.2.3
P10	Object-Oriented Notion of Context	theory graphs	4.3
P11	Framing	OMDoc views	4.3.2
P12	Recaps (via renamings)	views & adoptions	4.3.3
P13	Proofs	*not covered*	4.4
R1	Flexiformality	parallel markup at all levels	
R2	Pluralism	logics as theories	4.3.1
R3	Underspecification	—	

Figure 3: Phenomena and their Realizations

4.1 Object Level

At the level of mathematical objects OMDoc uses the MathML markup language [ABC+10] for representing the structure of mathematical formulae.

4.1.1 Mixing Formulae and Text

For phrase markup, OMDoc imports the XHTML5 [BFL+12, Chapter 9] inline (text) model, which supports embedded MathML formulae. As linguistic markup is highly theory dependent, we use the XHTML `span` element for word and sentence tokenization and employ RDF-based standoff markup whose subjects are XPath-encoded ranges and whose verbs come from annotation ontologies, e.g. from the OLiA framework [OLi]. As the URIs used in the RDF triples apply equally to XHTML5 elements, we have taken care of **P1** and **P2**.

MathML natively allows the flexible mixing of content and presentation representations, and even text. Consider the mixed formula $\{x \in \mathbb{N} | \ x \text{ is prime}\}$ from (1), which can be represented as

[6] The XML examples use the namespace prefixes o:, m:, and h: for OMDoc, MathML, and XHTML respectively.

shown in Figure 4. Following [ABC+10] we represent text in formulae as (presentation) operators, which makes the case for presentation MathML (on the left in Figure 4) very immediate. For content MathML, we make use of the fact that we can embed presentation MathML into content MathML token elements and embed the operator "*is prime*" via a m:mo element in functional position. The case for (3) is similar. Note the use of the m:csymbol element, which embodies

presentation MathML

```
<m:mrow>
  <m:mo>{</m:mo>
  <m:mrow>
    <m:mi>x</m:mo>
    <m:mo>∈</m:mo>
    <m:mo>ℕ</m:mo></m:mrow>
  <m:mo>|</m:mo>
  <m:mrow>
    <m:mi>x</m:mi>
    <m:mo> is prime</m:mo>
  </m:mrow>
  <m:mo>}</m:mo>
</m:mrow>
```

content MathML

```
<m:bind>
  <m:apply>
    <m:csymbol cd="set">setst</m:csymbol>
    <m:csymbol cd="nats">naturalnumbers</m:csymbol>
  </m:apply>
  <m:bvar><m:ci>x</m:ci></m:bvar>
  <m:apply>
    <m:symbol>is prime</m:csymbol>
    <m:ci>x</m:ci>
  </m:apply>
</m:bind>
```

Figure 4: Representing $\{x \in \mathbb{N} | \ x \text{ is prime}\}$ in MathML

the referential theory of meaning underlying content MathML. It points to the definition of the concept "*set separation*" and thus fixes the meaning of the set separation operator setst (to the extent that the theory set does; see Sections 4.2 and 4.3). This is a direct realization of **P4**.

4.1.2 Representing Ambiguity by Parallel Markup

MathML also allows representing both the form and the function of mathematical formulae by using *parallel markup*. This is implemented by the m:semantics element, which has presentation as the first child and the content in subsequent m:annotation−xml children[7]; corresponding subtrees are identified by cross-references. We will disregard this technical level here and concentrate on the concepts; let us look at an well-known example: The (presentation) formula $f(a+b)$ can be interpreted in two ways: as the application of the function f to the sum $a+b$ or as the product of a scalar f with the sum $a+b$. Consequently, we can annotate the presentation tree of $f(a+b)$ with two content trees to arrive at the situation depicted in Figure 5.

This shows that MathML is well-suited for representing ambiguity in formulae (cf. **P6**). Note that we can use the m:ref element (represented by an empty box with a dotted arrow) to point to a content MathML subtree and thus share structure and avoid representational redundancy. But note as well that both parses and serializations tend to explode as ambiguities from different sources compound, so that the treatment of ambiguity may have to be combined with techniques of grammatical and semantic underspecification or packed forest representations.

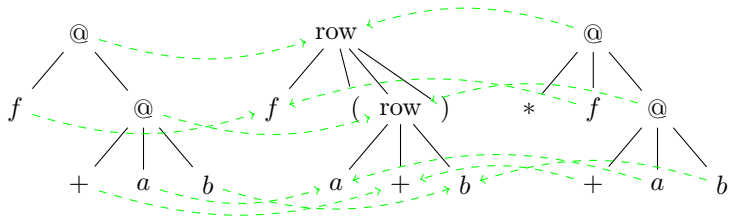

Figure 5: A presentation tree (center) with content annotations

4.1.3 Elision

[7] The other way around (content in the first child) is also possible, but not as suitable for our purposes.

For **P7** we note that since OMDoc is an XML language, we can just reconstruct (well-formed) sub-expressions from the context and mark them as elided by a special attribute. For instance, in the MathML representation of "$\log(x)$" on the right, we have marked the base of the logarithm with the elision level that ties in with the bracket elision level from [KMR08]: in the generated presentation MathML, the MathML element rendering of the base 2 would have CSS class elided500, which allows a user agent to reveal the base upon user request.

```
<m:apply>
  <m:csymbol cd="arith">log</m:csymbol>
  <m:mn o:elevel="500">2</m:mn>
  <m:mi>x</m:mi>
</m:apply>
```

But using parallel markup, we can do even better. Say we have the sequence "$1, 4, 9, \ldots, 81$", we can represent this as the parallel-markup tree in Figure 6.

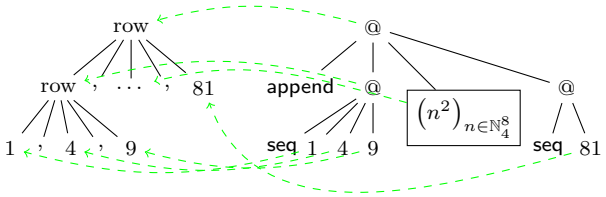

Figure 6: Specifying the content of an elision

There the content tree is the application of a function append to three sequences: the first is the initial sequence "$1, 4, 9$", the second is explicitly specified as the sequence of n^2, where n ranges over $\mathbb{N}_4^8$: the natural numbers from 4 to 8, and the third is the last value "81".[8]. Here, the linguistic elision marker "..." is linked to the sequence corresponding to its meaning. With parallel markup techniques like these, we can handle all elision techniques in mathematical practice; see [SS06] for some extreme examples of elision in matrixes.

4.1.4 Declarations

Declarations get phrase-level markup in OMDoc via a dedicated role system, which allows to designate a phrase (verbal or formula) as a declaration (via o:role="declaration"[9]). Additional attributes o:decname, o:type, o:restriction, o:quant, o:force, o:deps, and o:scope can be used to point to the respective functional parts of the declaration. Consider for instance the declaration "$\delta \in \mathbb{R}^+$" from (10). This would be marked up via the declaration attributes as shown in Listing 1. Here we have an existential declaration for the identifier δ[10] – designated by referencing its identifier del – of type $\mathbb{R}^+$ without further restriction.

Listing 1: A formulaic declaration

```
<m:apply o:role="declaration" o:force="existential" o:decname="#del" type="#pr" o:deps="#eps">
   <m:csymbol cd="set">in</m:csymbol>
   <m:ci id="del">δ</m:ci>
   <m:csymbol cd="reals" id="pr">positivereals</m:csymbol>
 </m:apply>
```

For the declaration "$x, y \in \mathbb{R}$ *with* $|x - y| \leq \delta$" in (10) we obtain the OMDoc markup in Listing 2, where we use a list of references in decname to point to the two identifiers declared here. Furthermore, we point to the explicit restriction "$|x - y| \leq \delta$".

Listing 2: A declaration with restriction

```
<h:span id="p173" o:role="declaration" o:force="universal"
                  o:decname="#x #y" o:type="#r" o:restriction="#rest">
  <m:apply o:role="declaration" o:force="universal" o:decname="#x #y" type="#r">
    <m:csymbol cd="set">minset</m:csymbol>
    <m:csymbol cd="reals" id="r">reals</m:csymbol>
    <m:ci id="x">x</m:ci>
    <m:ci id="y">y</m:ci>
```

[8] here and in the future we will use a boxed mathematical formula to represent a suitable MathML encoding that would have been too big for showing explicitly.

[9] Note that XHTML and MathML (and most other XML vocabularies) allow foreign-namespace attributes and ignore them.

[10] The force determined by the context of "$\delta \in \mathbb{R}^+$" in (10), but it should be considered a trait of the declaration. Furthermore, an existential declaration can depend on universal identifiers in the context, (here ε); this should also be recorded with the declaration.

```
  </m:apply>
  with
  <m:apply id="rest">
    <m:csymbol cd="numbers-orders">leq</m:csymbol>
    <m:apply>
      <m:csymbol name="arith">abs</m:csymbol>
      <m:apply>
        <m:csymbol name="arith">minus</m:csymbol>
        <m:ci id="x">x</m:ci>
        <m:ci id="y">y</m:ci>
      </m:apply>
    </m:apply>
    <m:ci>δ</m:ci>
  </m:apply>
</h:span>
```

Note that in this example the linguistic tokenization and the semantic visibility ranges coincide and comply with the XML tree structure. Without claiming universal coverage we observe that we have been able to arrange this so in all examples we studied.

For the declaration "$n > 10$ *is prime*" in (11) the situation is more complex: we have a identifier n which is restricted to be greater than 10 and a prime number – we can only point to the whole phrase in o:restriction as the CML uses a highly aggregated form.

Listing 3: A mixed verbal/formula declaration

```
<h:span id="p173" o:role="declaration" o:decname="n" o:scope="#s17 #s18" o:restriction="#p173"
        o:force="universal", quant="http://mathhub.info/nls/DRT?drt-base?delta">
  <m:apply>
    <m:csymbol cd="numbers-orders">greater</m:csymbol>
    <m:ci id="n">n</m:ci>
    <m:cn>10</m:cn>
  </m:apply>
  <h:span> is prime</h:span>
</h:span>
```

Note that this representation is truly flexiformal, it mixes informal, semantically opaque elements like "*is prime*" with formal ones (the content MathML markup for $n > 10$). Following [Bau99] and [Zin04] n would probably best be represented as a discourse referent in a DRT-like setting – after all (11) is really a donkey sentence – which is accessible in the two sentences of (11) (which we assume to have identifiers s17 and s18). We can even augment the force by marking up the particular quantifier via its Mmt URI. In essence, we delegate the specification of the dynamic behavior to the meta-level – see the introduction to Section 4.3 and 4.3.1 for a brief overview of OMDoc/Mmt and a discussion of pluralism.

4.2 Statement Level

OMDoc supplies its own markup infrastructure for making explicit the structure of mathematical statements (paragraphs that state properties about mathematical objects and their relations). This allows to categorize paragraphs into definitions, assertions (theorems, lemmata, conjectures,...), axioms, and proofs and make explicit their relations.

4.2.1 Dynamic Vocabulary and Symbolism

One of the foremost characteristics of the statement level in OMDoc is that it introduces the vocabulary and symbolism that make up the formulae and CML at the object level. Consider (12) that would typically be found in a definition statement. It introduces two symbolic identifiers: the (global) function name d for the depth function about to be defined and the (local) name T for the argument in question. In OMDoc we can use the markup below:

Listing 4: Introducing Vocabulary and Symbolism for tree depth – see (12)

```
<o:symbol name="tdepth"/>
<o:notation for="tdepth">
  <o:prototype><m:apply><m:csymbol>tdepth</m:csymbol><o:expr name="arg"/></m:apply></m:prototype>
  <o:rendering>
```

```
    <m:mrow><m:mo>d</m:mo><m:mo>(</m:mo><o:render name="arg"/><m:mo>)</m:mo></m:mrow>
  </o:rendering>
</o:notation>
<o:definition for="tdepth" xml:id="tdepth.def">
  <o:CMP><h:p>The <o:term role="definiendum" name="tdepth">depth</o:term>
    <m:apply><m:csymbol>tdepth</m:csymbol><m:ci>T</m:ci></m:apply>
    <o:term role="defnarg">of <o:term role="declaration">a tree <m:ci>T</m:ci></o:term></o:term>
  ...</h:p></o:CMP>
</o:definition>
```

As the concept of tree depth is global, we reserve the system identifier tdepth[11] using the OMDoc m:symbol element.

OMDoc notation definitions (the o:notation element) are a device for presenting content formulae. They consist of pairs of formula schemata, one in content MathML – the **prototype** – used to match a content MathML expression, and render it as the presentation MathML expression in the **rendering**[12]; the instances of any o:expr elements are rendered recursively. Note that the matching-based setup allows to distinguish 'applied' occurrences for functional objects; see [KMR08] for details. In our example the notation definition in lines 2 to 7 of Listing 4 introduces the formula notation for tree depth.

The verbalization for the symbol tdepth is marked up with the o:term element: The first occurrence of the relational noun "*depth*" is marked as a 'definiendum', i.e. as the defining occurrence of the noun (the definiens has been elided in this example). The phrase "*of a term* T" is characterized as an argument for the definiens "*depth*" by giving it the role="defnarg" and the phrase "*a tree* T" as a post-declaration for the identifier T. Note that we do not use a o:symbol element here, since T is definition-local. Also note that the notation and verbalization for the concept of tree depth are synchronized – think parallel markup at the statement level – by referencing the symbol identifier tdepth.

4.2.2 Statement-Level Parallel Markup

For statements, we have a similar content/form distinction as for mathematical formulae. In CML form, they take the form of suitably decorated paragraphs. From a foundational content perspective statements are simply triples of the form $c\colon \tau = \delta$, where c is a name (of a constant), τ is its type, and δ is its definiens – the latter two are optional. In Figure 7 we have an example of a definition. But a theorem would be very similar: via the Curry-Howard isomorphism – i.e. interpreting propositions as types given a sufficiently expressive type system – we can interpret a constant c of type $\tau := ded(\mathbf{A})$ ("proof of $\mathbf{A}$") as an axiom and a definition for c (i.e. by an expression δ of type $ded(\mathbf{A})$) as a proof.

On the left side of Figure 7, we have parallel markup between a narrative representation of a definition for a constant with symbol identifier foop by a text fragment ⟪text1⟫ in a o:CMP element, which is annotated with two (for the sake of argument) formalizations in o:definiens elements. Note that this setup is directly parallel to the one in Figure 5, where a presentation MathML formula is annotated with two content MathML expressions that formalize it.

On the right side of Figure 7 we see the dual situation, where a formal definition via a o:constant element is annotated by two narrative (definition) texts ⟪text1⟫ and ⟪text2⟫. In this way we can start with a narrative statement and formalize it in place by annotating it with "content OMDoc" or start with formal content (e.g. from a theorem prover) and annotate it with generated "presentation OMDoc" for human consumption. As in Section 4.1.1 we can freely interleave formal and informal parts, and as in Section 4.1.2 we can use parallel markup to represent ambiguity. Note that we do not need an OMDoc2 analogue to the generic semantics element in MathML, since the statement-level elements in OMDoc and Mmt already have the grouping capabilities needed to form a single root as required by XML.

[11] Note that this is only a system identifier for referencing the symbol, here we used the string tdepth to fortify the observation that this choice does not influence the verbalization or notation of the symbol.

[12] We can also use them the other way around for notation-definition driven parsing, matching presentation MathML formulae view the o:rendering element and constructing content MathML formulae via the o:prototype element.

narrative	content
<o:definition name="foop"> <o:meta rel="o:verbalizes" **resource**="?fooc"/> <o:CMP xml:id="foo.p">⟪text1⟫</o:CMP> <o:definiens xml:id="foo.c1"> <o:meta rel="o:formalizes" **resource**="#foo.p"/> $\boxed{\delta_1}$ </o:definiens> <o:definiens xml:id="foo.c2"> <o:meta rel="o:formalizes" **resource**="#foo.p"/> $\boxed{\delta_1}$ </o:definiens> </o:definition>	<o:constant name="fooc"> <o:meta rel="o:formalizes" **resource**="?foop"/> <o:definiens xml:id="foo.c">$\boxed{\delta}$</o:definiens> <o:CMP xml:id="foo.p1"> <o:meta rel="o:verbalizes" **resource**="#foo.c"/> ⟪text1⟫ </o:CMP> <o:CMP xml:id="foo.p2"> <o:meta rel="o:verbalizes" **resource**="#foo.c"/> ⟪text2⟫ </o:CMP> </o:constant>

Figure 7: A Simple Definition

4.2.3 Inline Statements

We can extend the idea of statement-level parallel markup directly to inline statements if we allow the parallel markup relations o:verbalizes and o:formalizes on the element h:span used for phrase markup in OMDoc. Consider for instance the situation of the theorem (13), which contains an embedded (inline) definition of the exponential function.

<o:assertion **type**="Theorem" o:reformulates="exp.thm">
<o:CMP>There is exactly one solution
(<h:span o:role="definition" o:reformulates="exp.def">the
<h:span o:role="definiendum">exponential function</h:span>
<h:span o:role="definiens">$\boxed{f(x) = e^x}$</h:span>
</h:span>)
to the equation $\boxed{f' = f}$ with $\boxed{f(0) = 1}$.
</o:CMP>
</o:assertion>

Here we classify the inline definition by o:role="definition" and mark up definiendum and definiens as usual. The reformulation relation with the expanded version from (14) and (15) is given by the o:reformulates attributes on the theorem and the inline definition.

<o:assertion **type**="Theorem" name="exp.thm">
<o:CMP>The equation $\boxed{f' = f}$ has exactly one solution with $\boxed{f(0) = 1}$.</o:CMP>
</o:assertion>
<o:definition name="exp.def">
<o:CMP>We call the function $\boxed{f}$ with $\boxed{f' = f}$ and $\boxed{f(0) = 1}$ the
<h:span role="definiens">exponential function</h:span> and write $\boxed{f(x) = e^x}$.
</o:CMP>
</o:definition>

4.3 Theory Level

At the theory level, OMDoc provides its own markup for organizing sets of statements into *theories*, and for specifying relations between them by *views*. This permits structuring mathematical knowledge into reusable chunks. Theories also serve as the primary notion of context in OMDoc, so they are the natural target for the context aspect of formula and statement markup. Organizing symbols into theories also limits their scope and permits a finer organization of knowledge.

Figure 8 shows an example OMDoc theory graph containing theories for Monoid, CGroup (Commutative Group), Ring and Integers. We will employ the OMDoc/Mmt system [RK13], a rational re-development of the theory level of OMDoc, which greatly enhances the modular aspects and clarifies the properties of theories and views that are central to this section.

In OMDoc *imports* are special statements that may occur in a theory S to permit symbols from another theory T to be used in S. Imports may be complete (import all symbols from T) or partial (import only some selected symbols) and may be direct (called *include* in Mmt) – they import symbols as they are – or indirect (called *structures* in Mmt) – they import via a named translation function, e.g. renaming symbols or translating objects over a view.

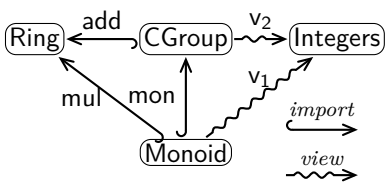

Figure 8: OMDoc Theory Graph

The intuition behind views is that they formalize theory translations by representing one theory in terms of another (e.g. integers as a group). A view from theory S to theory T interprets (some of) the symbols of S in terms of symbols in T with the intuition that their properties are preserved. In the formal case, it can be a formal translation that assigns a term in T to every symbol in S while preserving structure and typing (as in [RK13]), in the informal cases it can be a textual description.

Consistent with mathematical intuitions the theory graph from Figure 8 defines CGroup by including Monoid and forms Ring by two indirect imports: add from CGroup for addition and mul from Monoid for multiplication. The relations between integers and monoids and groups are formalized by the views v_1 and v_2.

Such views are essential for representing *dynamic vocabulary* (**P10**), *framing* (**P11**) and, together with imports, *reccaps via renamings* (**P12**); see below for details.

Each OMDoc object has a canonical identifier: its Mmt URI; see [RK13]. Mmt URIs conform to the URI grammar given in RFC 3986 [BLFM05] and can be *i*) OMDoc **document identifiers** – URIs without queries or fragments, *ii*) **Module identifiers** $\mathtt{T} = \mathtt{D?t}$ – formed by pairing a document identifier $\mathtt{D}$ with a theory name $\mathtt{t}$, separated by the character "?"), or *iii*) **statement identifiers** $\mathtt{S} = \mathtt{T?s_r}$ – formed from a theory identifier $\mathtt{T}$ and statement reference $\mathtt{s_r}$. Statement references are of the form $\mathtt{s_r} = \mathtt{v_1/\ldots/v_n/s}$ where $\mathtt{v_i}$ are views inducing the statement $\mathtt{s}$ into T. For example, the theory Ring in Figure 8 can access the symbols add/mon/comp (addition), add/mon/unit (additive unit), add/inv (additive inverse), mul/comp (multiplication), and mul/unit (multiplicative unit).

4.3.1 Logical Pluralism

OMDoc/Mmt adopts *heterogeneous pluralism* by allowing the representation of logics and logical frameworks as theories. Then, other theories can build on them, by importing the relevant logic or logical framework. In order to more precisely express the relation between logical framework and logic or logic and theory, OMDoc provides the meta relation as a distinguished kind of import.

Using views, translations at the logical or logical-framework level, induce translation at the lower levels allowing one to relate different logical languages and make use of that by, e.g. automatically translating documents from one logic to another.

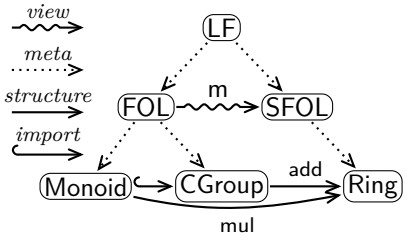

Figure 9: Pluralism in OMDoc

Figure 9 (extending the example in Figure 8) shows an instance of logical pluralism in OMDoc/Mmt where the logical framework LF and the logics FOL (First-Order Logic) and SFOL (Sorted First-Order Logic) are declared as theories. Then, Monoid and Cgroup build on FOL, while Ring – for the sake of argument – has SFOL as a meta-theory. The relation between FOL and SFOL is formalized as a view, inducing a translation of Ring and Monoid into SFOL, which gives meaning to the two structures add and mul that make up most of the content of Ring.

4.3.2 Understanding Framing

OMDoc can directly interpret framing as the application of a view in a theory graph: the operation of flattening (copying/translating all axioms and concepts from the source theory of the view to

the target theory; see [RK13] for a discussion) directly accounts for the bridging references in (19) and (20). In a sense, supporting framing as a mathematical practice has been a primary goal of the OMDoc format from the start, here we are only applying it to linguistics.

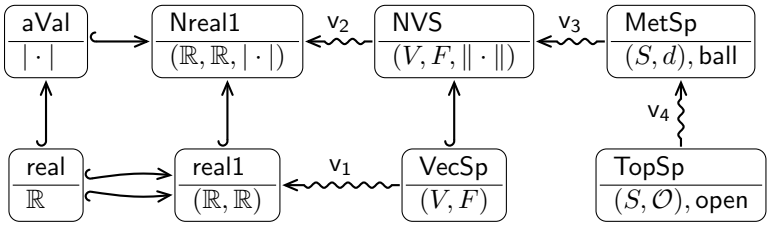

Figure 10: Framing Spaces

Consider for instance the situation in example (20), where we have the absolute value $|\cdot|$ on $\mathbb{R}$, which, together with the fact that $\mathbb{R}$ forms a one-dimensional vector space over the field of real numbers (witnessed by view v_1), makes NReal1 into a normed vector space (see view v_2). Furthermore, we have a view v_4 from the theory TopSp of topological spaces to the theory MetSp of metric spaces: the topology $\mathcal{O}$ over a set S can be defined as the collection of unions of open balls $\mathsf{ball}(s,r)$ with $s \in S$ and $r \in \mathbb{R}$, where $\mathsf{ball}(s,r) := \{t \in S | d(s,t) < r\}$. Finally there is a view v_3 from MetSp to the theory NVS of normed vector spaces: we can define a distance function d by $d(x,y) := \|x-y\|$. In this situation, flattening supplies the concepts of "*ball*" and "*open*" to NVS, accounting for the bridging references in (20).

For (19) we need to explain the application of the adjective "*discrete*" to the complex noun "*normed vector space*". Recall that a metric space is called discrete, iff its distance function is ($d(x,y) = 0$, iff $x = y$, else $d(x,y) = 1$), so the adjective "*discrete*" subcategorizes for structures that have a distance function, which NVS does thanks to the view v_3.

4.3.3 Modeling Recaps

Following the theory graph approach outlined above, we structure the main part of a standalone document in a graph of (new) theories that introduce the payload knowledge of that document. Usually, the payload directly or indirectly builds on concepts and results from the literature. We can model those concepts that are directly used via direct inclusions of external theories. In contrast to this, recaps often function as overrides, changing notations and sometimes even definitions and theorems (typically to simplify, or restrict to the relevant cases). In that regard, recaps can be seen as linguistic expressions of a complex notion of inheritance or importing: In Figure 11 we model the situation in (21): we have a careful development of the elementary algebraic hierarchy in four theories, and two structures (imports via renamings) on the lower left. The theory ring (and thus the theories it depends upon) is adopted into the theory AlgRecap that contains flexiformalization of (21) via a postulated view from AlgRecap to ring (i.e. all concepts of ring are postulated to be true in AlgRecap) which induces a partial inclusion from ring into AlgRecap (the very intention of the construction) on which the payload theories can build.

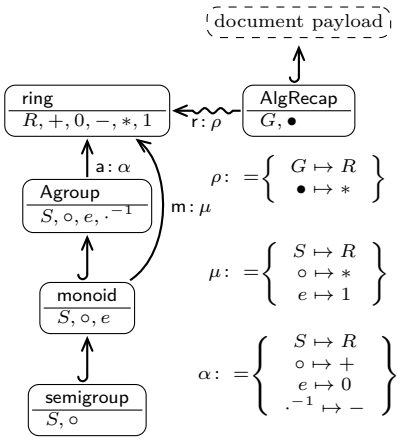

Figure 11: Recap by Adopting a Background Theory Graph

The case of the recap (22) of the definition (23) is more difficult, since we do not have a direct generalization relation as in the case above. In this case, the definition of the accelerated Turing machine involves a concrete step size (2^{-n}), whereas the definition it recaps allows arbitrary sequences of step sizes as long as their sum remains finite. Thus we have the situation in Figure 12. Theory ATM contains the (opaque) sentence (22), but there cannot be a view from ATM to atm as that is more general. But we do have a view to $\mathsf{atm}(2^{-\mathsf{n}})$, which naturally arises in treatments of accelerated Turing machines as an example. The more important aspect is that the contribution

of [CS09] (depicted here as theory ATMhalt as it concerns undecidability of the accelerated halting problem by accelerated Turing machines) can be justified via a view into a theory $\mathsf{atm}(2^{-n})\mathsf{halt}$, which can be systematically constructed from ATMhalt modulo v_1 as pushouts along inclusions always exist in OMDoc/Mmt.

In the particular example, we can do even better: as the proof in theory ATMhalt does not use any properties of the step size sequence 2^{-n}, there is a theory atmhalt that generalizes $\mathsf{atm}(2^{-n})\mathsf{halt}$ (shown dotted in Figure 12). Note that this construction is implicit and needs human intervention to infer – but one that the authors expect their readers will readily do, otherwise they would not have restricted themselves to the concrete sequence.

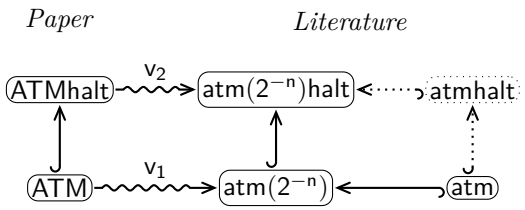

Figure 12: Recaps for ATM

4.4 Not covered in this Article: Proofs

OMDoc has native markup for proofs that accommodates many of the phenomena alluded to in Section 2.4, but we will not detail this here, and refer the reader to [Koh06b, chapter 17] and leave a linguistic/semantic evaluation to future work; a semantic evaluation of the representational adequacy in terms of formal proof formats of proof checkers has appeared in [ASC06, SC10].

5 Related Work

We have seen that OMDoc can adequately represent many of the phenomena presented in Section 2, let us compare it with others in the space of representation formats for mathematical knowledge; Figure 13 gives an overview.

	FOL	Naproche	WTT	MathLang	OMDoc
P1 Formulae as Linguistic Objects	+	+	+	+	+
P2 Formulae as Grammatical Objects	+	+	+	+	+
P3 Notations and Verbalizations	−	−	−	+	+
P4 Referential Meaning	+	+	+	+	+
P5 Declarations: Naming Objects Locally	+	+	+	+	+
P6 Ambiguity	−	0	+	+	+
P7 Elision	−	−	−	+	+
P8 Mathematical Statements	−	+	+	+	+
P9 Inline Statements	−	−	−	+	+
P10 Object-Oriented Notion of Context	−	−	0	0	+
P11 Framing	−	−	−	−	+
P12 Recaps (via renamings)	−	−	−	−	+
R1 Flexiformality	−	0	0	+	+
R2 Pluralism	−	−	0	0	+
R3 Underspecification	−	−	−	−	−

Figure 13: Features of Mathematical Knowledge Representation Languages

First-Order Logic On the surface, first-order logic is an appealing target format for translating mathematical documents since, together with axiomatic set theory [Ber91], it forms the most widely accepted foundation of mathematics. However, due to its first-order nature and limitations on handling some natural language phenomena, (e.g. anaphora) other variants, such as higher-order or dynamic logics [Har84] were identified as more appealing.

In general, FOL and related logics are good at representing formulae (**P1**:+, **P2**:+), and allow for named declarations (**P5**:+) and references (**P4**:+). However, it is not realistically possible to represent notations and verbalizations (**P3**:-) as well as to handle ambiguity (**P6**:-) or elision (**P7**:-).

Moreover, while there are many ways to extend FOL with a more or less complex module system, modularity is not typically a part of FOL's standard formulation. Therefore, one cannot adequately represent the high level discourse (**P8**:-, **P9**:-) and context (**P10**:-, **P11**:-, **P12**:-) structure of mathematical documents (e.g. theorems, definitions, proofs, theories, etc. and the relations between them). As a direct consequence, the phenomena from the discourse and context level discussed in Section 2 cannot be captured in plain FOL or related logics.

Naproche The Naproche project [CFK+10] starts from the common mathematical language to develop a *controlled natural language* (CNL) for mathematical texts (**R1**:0) and a proof checking software to formally check texts written in this language. Therefore, the Naproche CNL focuses on proofs to permit adequate proof checking by the Naproche system and doesn't necessarily aim to cover all phenomena from Section 2. Still, it is interesting to evaluate it with respect to these requirements.

At the phrase level, the Naproche CNL includes a language for formulae which can be integrated within natural language phrases (**P1**:+, **P2**:+) and include references (**P4**:+) and declarations (**P5**:+). However, there is no explicit support for notations and verbalizations (**P3**:-) or elisions (**P7**:-) and only limited coverage of ambiguity (as one would expect from a controlled language; **P6**:0)

At the discourse level, Naproche's CNL offers explicit markup for *axioms*, *definitions*, *lemmas* and *theorems* as well as *proofs* with *case distinctions* and *assumptions* (**P8**:+) but not for inline statements (**P9**:-). Proofs are represented using *proof representation structures* (PRS) which are adapted from discourse representation structures [KR93].

The context level, as defined in 2, is missing as there is no explicit markup for documents, theories or groups of statements (**P10**:-, **P11**:-, **P12**:-).

Weak Type Theory WTT (Weak Type Theory) [KN04] is a refinement of de Bruijn's Mathematical Vernacular (**R1**:0) and is designed to act as an intermediary between common mathematical language and formal mathematics based on various logics (**R2**:0).

At the *phrase level*, WTT has primitives inspired by common mathematical language: terms, sets, nouns and adjectives (**P1**:+, **P2**:+). At the statement level, WTT has *statements* and *definitions* (**P8**:+) but doesn't distinguish, for instance, examples or lemmas and does not allow for inline statements (**P9**:-). *Books* in WTT are the only context level element, roughly representing ordered collections of statements (**P10**:0). Therefore, the context level of WTT is limited in expressivity, for instance with respect to framing (**P11**:-) or complex knowledge sharing via parametric imports (**P12**:-).

As WTT is designed to be structurally close to the grammar of natural language it is suitable as a first step in the formalization process. Consequently, translating a CML text into WTT is significantly easier than fully formalizing it [Joj05, Gel04]. However, WTT is relatively weak and limited in terms of capturing the semantics of mathematical texts (especially at the context level) and further formalization can require significant effort [Joj05, Sch03].

MathLang Roughly based on WTT, MathLang [KWZ08] aims at reaching a compromise between expressivity and formality and thus provide an interface language between mathematicians

and computers (**R1**:+) as well as a framework to make the links with existing formal proof systems (**R2**:0).

With respect to semantization and knowledge representation MathLang adopts a paradigm of annotating (fragments of) texts with their grammatical or semantic category. Its design distinguishes three different levels of annotation granularity (called "aspects" in MathLang).

- The Core Grammar Aspect (CGa) is a kind of weak type system directly based on WTT (**P1**:+, **P2**:+, **P4**:+, **P5**:+, **P6**:+) [KN04] and de Bruijn's mathematical vernacular [dB87]. The types themselves are motivated by both mathematical and grammatical considerations, including e.g. "noun", "adjective", "set", "definition" or "context".

- The Text and Symbol Aspect (TSa) enables establishing the association between textual presentation and mathematical meaning. Although still annotation based, the effort to weave together presentation-oriented and content-oriented representations is reminiscent of MathML's parallel markup. A notion of *souring* rules (named in relation to syntactic *sugaring*) is introduced to permit better control over the structure of mathematical texts. As a result it allows representing notations (**P3**:+) and elisions (**P7**:+).

  Together, CGa and TSa mostly correspond to the OMDoc object level although some CGa categories such as "definition", "declaration" and "statement" straddle the border to the statement level.

- The Document Rhetorical Aspect (DRa) allows labeling fragments of text and establishing relations between them. For instance a text fragment can be labeled as a "theorem" and another as a "proof" for it. This corresponds roughly to the statement level in OMDoc allowing both top-level and inline statements (**P8**:+, **P9**:+).

Just like WTT, which it extends, MathLang lacks at the context level where it only has "documents" as ordered collections of statements (**P10**:0) and no support for framing (**P11**:-) or recaps (**P12**:-).

Homogenous Pluralism With the OMDoc format we have presented a representation that takes the concept of heterogeneous pluralism to the extreme, but there are also logics in the homogenous tradition that might be worth considering as bases for representation formats for mathematical linguistics: Type theories with dependent record types (e.g. the calculus of constructions (CoC [CH88]) of the Coq system [BC04]) can represent many of the structures at the theory level in a logic-internal way by encoding what we think of as theories as record types and views as type coersions. However, it is still unclear whether the CoC or its derivatives can cope with dynamic approaches like DRT or DPL, though Aarne Ranta's work [Ran94, Ran04] on GF might also be of relevance here.

6 Conclusion

In this article we have identified requirements for target representation formats for semantics extraction from mathematical languages: We carefully identified a set of challenging linguistic/semantic phenomena and discuss how they can be modeled in the OMDoc format. We show that the three-level content markup architecture fits well with the distribution of linguistic phenomena on the phrase, discourse, and context levels. In related study [Koh14a], we have shown that OMDoc/Mmt can be used as a data model for lexical resources for multilingual mathematical terminologies; there the theory graph model was used to account for terminological relations (hypernymy, metonymy, antonymy, etc.).

From this exercise, we can conclude that the (flat sets of) *first-order formulae*, which have been the "default" target format for both the foundations of mathematics and semantics construction in computational linguistics are *not an adequate target language for formalization/semantics extraction*. In summary, we notice that we need

i: a referential theory of meaning (by pointing to symbol definitions)
ii: three levels of representation (objects/statements/theories)
iii: parallel markup (mix formal/informal recursively at any level)
iv: flexiformality: to allow opaque content (presentation/natural language)
v: pluralism at all levels (object/logic/foundation/meta-logic)
vi: underspecification of symbol meaning

The work in OMDoc is a first step into the direction of designing a target format for the meaning of mathematical documents and knowledge in the large, but further research and format design efforts have to be undertaken for the last two aspects. We are currently working on an improved language design that takes conditions *i* to *vi* above as design goals and integrates Mmt into OMDoc as a logical core. Note that conditions *v* and *vi* are inter-dependent, pluralism allows to use logical systems that integrate methods for underspecification, and we can see pluralism at a step towards underspecification at the logical level: In the semantics of natural language, we should not have to commit to particular details of the logical foundation; we are just interested in "using" the logic for representation.

Finally, we observe that this article concentrates on establishing (criteria for) a target representation format for the semantics of CML and presents OMDoc as a relatively complete first approximation. The question of how to construct adequate OMDoc representations from mathematical documents is still largely unsolved.

Acknowledgements The work in this article has been partially supported by the Leibniz association under grant SAW-2012-FIZ_KA-2 and the German Research Foundation (DFG) under grants KO 2428/9-1 and KO 2428/13-1. The authors gratefully acknowledge the discussions with Deyan Ginev, Andrea Kohlhase, Manfred Pinkal, Florian Rabe, and Magdalena Wolska which led to a much better understanding of the concepts presented in this article.

References

[ABC+10] Ron Ausbrooks, Stephen Buswell, David Carlisle, Giorgi Chavchanidze, Stéphane Dalmas, Stan Devitt, Angel Diaz, Sam Dooley, Roger Hunter, Patrick Ion, Michael Kohlhase, Azzeddine Lazrek, Paul Libbrecht, Bruce Miller, Robert Miner, Murray Sargent, Bruce Smith, Neil Soiffer, Robert Sutor, and Stephen Watt. Mathematical Markup Language (MathML) version 3.0. W3C Recommendation, World Wide Web Consortium (W3C), 2010.

[ACR+08] Serge Autexier, John Campbell, Julio Rubio, Volker Sorge, Masakazu Suzuki, and Freek Wiedijk, editors. *Intelligent Computer Mathematics*, number 5144 in LNAI. Springer Verlag, 2008.

[ASC06] Serge Autexier and Claudio Sacerdoti Coen. A formal correspondence between omdoc with alternative proofs and the $\overline{\lambda}\mu\tilde{\mu}$-calculus. In Jon Borwein and William M. Farmer, editors, *Mathematical Knowledge Management (MKM)*, number 4108 in LNAI, pages 67–81. Springer Verlag, 2006.

[Bau99] Judith Baur. Syntax und Semantik mathematischer Texte — ein Prototyp. Master's thesis, Fachrichtung Computerlinguistik, Universität des Saarlandes, SaarbrückenGermany, 1999.

[BC04] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development — Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. An EATCS Series. Springer Verlag, 2004.

[Ber91] Paul Bernays. *Axiomatic Set Theory*. Dover Publications, 1991.

[BFL+12] Robin Berjon, Steve Faulkner, Travis Leithead, Erika Doyle Navara, Edward O'Connor, Silvia Pfeiffer, and Ian Hickson. HTML5. W3C Candidate Recommentation, World Wide Web Consortium (W3C), 2012.

[BLFM05] Tim Berners-Lee, Roy T. Fielding, and Larry Masinter. Uniform resource identifier (URI): Generic syntax. RFC 3986, Internet Engineering Task Force (IETF), 2005.

[BP64] Paul Benacerraf and Hilary Putnam, editors. *Philosophy of mathematics: Selected readings*. Cambridge University Press, 2nd edition 1983 edition, 1964.

[CFK+10] Marcos Cramer, Bernhard Fisseni, Peter Koepke, Daniel Kühlwein, Bernhard Schröder, and Jip Veldman. The Naproche project controlled natural language proof checking of mathematical texts. In Norbert E. Fuchs, editor, *Controlled Natural Language, Workshop on Controlled Natural Language, CNL 2009. Revised Papers*, number 5972 in LNCS, pages 170–186. Springer, 2010.

[CH88] Thierry Coquand and Gérard Huet. The Calculus of Constructions. *Information and Computation*, 76(2/3):95–120, 1988.

[CHK+11] Mihai Codescu, Fulya Horozal, Michael Kohlhase, Till Mossakowski, and Florian Rabe. Project abstract: Logic atlas and integrator (LATIN). In James Davenport, William Farmer, Florian Rabe, and Josef Urban, editors, *Intelligent Computer Mathematics*, number 6824 in LNAI, pages 289–291. Springer Verlag, 2011.

[CS09] Cris Calude and Ludwig Staiger. A note on accelerated turing machines. CDMTCS Research Report 350, Centre for Discrete Mathematics and Theoretical Computer Science, Auckland University, 2009.

[dB87] Nicolaas Govert de Bruijn. The mathematical vernacular, a language for mathematics with typed sets. In P. Dybjer et al., editors, *Proceedings of the Workshop on Programming Languages*, 1987.

[DT00] Marc Dymetman and Frédéric Tendeau. Context-free grammar rewriting and the transfer of packed linguistic representations. In *COLING 2000, 18th International Conference on Computational Linguistics*. Morgan Kaufmann, 2000.

[Gan13] Mohan Ganesalingam. *The Language of Mathematics, A Linguistic and Philosophical Investigation*, volume 7805 of *LNCS*. Springer Verlag, 2013.

[Gel04] Gijs Geleijnse. Comparing two user-friendly formal languages for mathematics: Weak type theory and mizar. Master's thesis, Technische Universiteit Eindhoven, 2004.

[Gin11] Deyan Ginev. The structure of mathematical expressions. Master's thesis, Jacobs University Bremen, Bremen, Germany, August 2011.

[GS90] Jeroen Groenendijk and Martin Stokhof. Dynamic Montague Grammar. In L. Kálmán and L. Pólos, editors, *Papers from the Second Symposium on Logic and Language*, pages 3–48. Akadémiai Kiadó, Budapest, 1990.

[GS91] Jeroen Groenendijk and Martin Stokhof. Dynamic predicate logic. *Linguistics & Philosophy*, 14:39–100, 1991.

[Har84] David Harel. Dynamic Logic. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume II, pages 497–604. Reidel, Dordrecht, 1984.

[JM09] Daniel Jurafsky and James H. Martin. *Speech And Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, second edition, 2009.

[Joj05] Gueorgui I. Jojgov. Translating a fragment of weak type theory into type theory with open terms. In Kohlhase [Koh06a], pages 389–403.

[KK09] Andrea Kohlhase and Michael Kohlhase. Spreadsheet interaction with frames: Exploring a mathematical practice. In Jacques Carette, Lucas Dixon, Claudio Sacerdoti Coen, and Stephen M. Watt, editors, *MKM/Calculemus Proceedings*, number 5625 in LNAI, pages 341–356. Springer Verlag, July 2009.

[KKP96] Michael Kohlhase, Susanna Kuschert, and Manfred Pinkal. A type-theoretic semantics for λ-DRT. In P. Dekker and M. Stokhof, editors, *Proceedings of the 10th Amsterdam Colloquium*, pages 479–498, Amsterdam, 1996. ILLC.

[KMR08] Michael Kohlhase, Christine Müller, and Florian Rabe. Notations for living mathematical documents. In Autexier et al. [ACR+08], pages 504–519.

[KN04] Fairouz Kamareddine and Rob Nederpelt. A refinement of de Bruijn's formal language of mathematics. *Logic, Language and Information*, 13(3):287–340, 2004.

[Koh06a] Michael Kohlhase, editor. *Mathematical Knowledge Management, MKM'05*, number 3863 in LNAI. Springer Verlag, 2006.

[Koh06b] Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, August 2006.

[Koh13] Michael Kohlhase. The flexiformalist manifesto. In Andrei Voronkov, Viorel Negru, Tetsuo Ida, Tudor Jebelean, Dana Petcu, Stephen M. Watt, and Daniela Zaharie, editors, *14th International Workshop on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2012)*, pages 30–36, Timisoara, Romania, 2013. IEEE Press.

[Koh14a] Michael Kohlhase. A data model and encoding for a semantic, multilingual glossary of mathematics. In Stephan Watt, James Davenport, Alan Sexton, Petr Sojka, and Josef Urban, editors, *Intelligent Computer Mathematics 2014*, Lecture Notes in Computer Science, pages 169–183. Springer, 2014. accepted.

[Koh14b] Michael Kohlhase. Mathematical knowledge management: Transcending the one-brain-barrier with theory graphs. *EMS Newsletter*, 2014. in press.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic: Introduction to Model-Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer, Dordrecht, 1993.

[KTP10] Alexander Koller, Stefan Thater, and Manfred Pinkal. Scope underspecification with tree descriptions: Theory and practice. In Matthew Crocker and Jörg Siekmann, editors, *Resource Adaptive Cognitive Processes*, Cognitive Technologies Series. Springer, Berlin, 2010.

[KWZ08] Fairouz Kamareddine, J. B. Wells, and Christoph Zengler. Computerising mathematical text with mathlang. *Electron. Notes Theor. Comput. Sci.*, 205:5–30, 2008.

[Mon74] Richard Montague. The proper treatment of quantification in ordinary English. In R. Thomason, editor, *Formal Philosophy. Selected Papers*. Yale University Press, New Haven, 1974.

[Mos05] Till Mossakowski. Heterogeneous specification and the heterogeneous tool set. Habilitation thesis, University of Bremen, 2005.

[OLi] OLiA ontologies. `http://nachhalt.sfb632.uni-potsdam.de/owl/`. seen Nov. 2013.

[Pfe01] Frank Pfenning. Logical frameworks. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, volume I and II. Elsevier Science and MIT Press, 2001.

[Pus98] James Pustejovsky. The semantics of lexical underspecification. *Folia Linguistica*, 32:323–347, 1998.

[Ran94] Aarne Ranta. Type theory and the informal language of mathematics. In H. Barendregt and T. Nipkow, editors, *Types for Proofs and Programs*, number 806 in LNCS, pages 352–365, 1994.

[Ran04] Aarne Ranta. Grammatical framework — a type-theoretical grammar formalism. *Journal of Functional Programming*, 14(2):145–189, 2004.

[RK13] Florian Rabe and Michael Kohlhase. *Information & Computation*, 0(230):1–54, 2013.

[SC10] Claudio Sacerdoti Coen. Declarative representation of proof terms. *Journal of Automated Reasoning; Special Issue on Programming Languages for Mechanized Mathematical Systems*, 44:25–52, 2010.

[Sch03] Mark Scheffer. Formalizing mathematics using Weak Type Theory. Master's thesis, Eindhoven University of Technology, 2003.

[SK08] Heinrich Stamerjohanns and Michael Kohlhase. Transforming the arXiv to XML. In Autexier et al. [ACR+08], pages 574–582.

[SS06] Alan Sexton and Volker Sorge. Processing textbook-style matrices. In Kohlhase [Koh06a], pages 111–125.

[WG10] Magdalena Wolska and Mihai Grigore. Symbol declarations in mathematical writing: A corpus study. In Petr Sojka, editor, *Towards Digital Mathematics Library, DML workshop*, pages 119–127. Masaryk University, Brno, 2010.

[WGK11] Magdalena Wolska, Mihai Grigore, and Michael Kohlhase. Using discourse context to interpret object-denoting mathematical expressions. In Petr Sojka, editor, *Towards Digital Mathematics Library, DML workshop*, pages 85–101. Masaryk University, Brno, 2011.

[Wol12] Magdalena Wolska. Building a pos-annotated corpus of scientific papers in mathematics. In Petr Sojka and Michael Kohlhase, editors, *DML and MIR 2012*. Masaryk University, Brno, 2012. in press.

[Wol13] Magdalena A. Wolska. *Student's Language in Computer-Assisted Tutoring of Mathematical Proofs*. PhD thesis, ComputerLinguistik, Saarland University, 2013.

[Zin04] Claus Zinn. *Understanding Informal Mathematical Discourse*. PhD thesis, Technischen Fakultät der Universität Erlangen-Nürnberg, 2004.