A Formal Correspondence between OMDoc with Alternative Proofs and the $\overline{\lambda}\mu\tilde{\mu}$-Calculus

Serge Autexier[1] and Claudio Sacerdoti-Coen[2]

[1] DFKI GmbH and Saarland University, Saarbrücken, Germany, `autexier@dfki.de`
[2] Department of Computer Science, University of Bologna, Italy, `sacerdot@cs.unibo.it`

Abstract. We consider an extension of OMDoc proofs with alternative sub-proofs and proofs at different level of detail, and an affine non-deterministic fragment of the $\overline{\lambda}\mu\tilde{\mu}$-calculus seen as a proof format. We provide explanations in pseudo-natural language of proofs in both formats, and a formal correspondence between the two by means of two mutually inverse encodings of one format in the other one.

1 Introduction

Proofs play a major role in mathematics and their representation is a key issue in mathematical knowledge management. Proofs of all kinds need to be stored and retrieved in repositories of formalized mathematics and communicated across system and logic boundaries, for instance to be assembled to larger proofs in the context of computer assisted mathematical theorem proving or to be explained on an adaptive level of granularity in tutor systems for teaching mathematics.

In [1] the first author and his colleagues presented a data structure for the representation of proof attempts (proof data structure or *PDS*). The two main features of a PDS are the possibility of representing proofs at different levels of granularity and that of representing alternative, possibly incomplete, subproofs. A PDS is quite complex, being a directed graph with nodes representing proof goals, arcs justifying a proof goal with some subgoals via a calculus rule, high-level proof method or a proof sketch and *hierarchical arcs* that represent transitions between granularity levels. There can be more than one justification for each node, which are at different levels of granularity if connected by hierarchical arcs and alternative subproofs otherwise. A PDS maintains simultaneously all proofs at different levels of granularity as well as all alternative proofs of some subgoals, which are useful features during interactive or automatic proof construction, and also for proof explanation, for instance in a tutorial setting. Selection of a specific level of granularity is supported by *views* on a PDS. In this paper we consider *views* to also include the selection of exactly one alternative subproof for each goal. To store and communicate the proofs and proof plans contained in a PDS, a PDS can be serialized to a proof format, that can be OMDoc [3]. However, OMDoc is currently unable to represent alternative subproofs.

Both PDSs and OMDoc documents claim to be adequate representation formalisms for mathematical proofs. However, they lack a semantics specification

Terms	*Environments*	*Reduction rules* :
$v ::= x$	$E ::= \alpha$	$\langle \mu\alpha : T.c \| E\rangle \quad \rhd c\{E/\alpha\}$
$\mid \lambda x : T.v$	$\mid v \circ E$	$\langle v \| \tilde{\mu}x : T.c\rangle \quad \rhd c\{v/x\}$
$\mid \mu\alpha : T.c$	$\mid \tilde{\mu}x : T.c$	$\langle \lambda x : T.v_1 \| v_2 \circ E\rangle \rhd \langle v_2 \| \tilde{\mu}x : T.v_1 \circ E\rangle$
	Commands	η-like rules:
	$c ::= \langle v \| E\rangle$	μ-expansion: $v \Rightarrow \mu\alpha : T.\langle v \| \alpha\rangle$
		$\tilde{\mu}$-expansion: $E \Rightarrow \tilde{\mu}x : T.\langle x \| E\rangle$

Table 1. Syntax of the $\overline{\lambda}\mu\tilde{\mu}$-calculus & $\overline{\lambda}\mu\tilde{\mu}$-calculus reduction rules

and are so generic that any structured document can be embedded into them. To make more precise the semantics of OMDoc and other proof formats, the second author has started the investigation of the $\overline{\lambda}\mu\tilde{\mu}$ calculus as a proof format in [4]. The calculus, that is Curry-Howard isomorphic to classical sequent calculus, has very remarkable properties per se and also as a proof format. In particular, as explained in [4], its intuitionistic (and deterministic) fragment can be equipped with a straightforward translation to pseudo-natural language, and proofs in the classical fragment can be easily translated to the intuitionistic fragment. What was so far unclear is whether the non-deterministic classical fragment of the calculus (or sub-fragments of it) is necessary to fully exploit the calculus as a proof format. In this paper we answer positively the question by proposing an extension of OMDoc with alternative proofs (inspired by PDSs) and an encoding of the extension in a fragment of the $\overline{\lambda}\mu\tilde{\mu}$-calculus. The encoding provides naturally a semantics for a view: a view on a proof is obtained by reducing the corresponding proof term according to the non-deterministic rules of the calculus. The non determinism dynamically selects just one of the alternative proofs for each choice. The encoding is particularly informative for two reasons. First of all it provides a clear semantics for OMDoc (and, indirectly, for the corresponding PDSs). Secondly it tries to respect the *rendering semantics* associated to OMDoc and to the $\overline{\lambda}\mu\tilde{\mu}$-calculus. A rendering semantics is the function that translates the term to its pseudo-natural language rendering.

2 $\overline{\lambda}\mu\tilde{\mu}$-Calculus

Table 1 shows the syntax of the $\overline{\lambda}\mu\tilde{\mu}$-calculus, proposed by Curien and Herbelin in [2]. Its rendering semantics can be found in [4]. In the rest of the paper we assume the reader to be familiar with the latter paper, while knowledge of the first one is not necessary.

The *intuitionistic* fragment of the calculus is obtained by restricting the set of continuation variables (ranged over by Greek letters) to a singleton (whose only element is conventionally the $\star$ symbol). This way every time a continuation is bound by the μ binder the previous bound continuation goes out of scope. The rendering semantics in [4] provides an easy intuition: every time we state that we are going to prove something we are obliged to conclude it; in other words, subproofs must be well-nested. The intuitionistic fragment is Curry-Howard isomorphic to Gentzen LJ sequent calculus.

A term that is not in the intuitionistic fragment is said to be in the *classical* fragment of the calculus. In this fragment it is possible to bound a continuation, but later on give control to another continuation bound less recently in the past. The rendering semantics in [4] provides an easy intuition, but the result is not at all natural: we can state that we are going to prove something, but later on we can escape to an outer proof and conclude it instead. In doing this we can exploit the additional hypotheses we collected in the inner (and unfinished) proof. Since subproofs are not well-nested in this fragment, the pseudo-natural language obtained is not natural nor easy to understand. For this reason in [4] we apply the rendering semantics only to the intuitionistic fragment and provide a translation from the classical fragment to the intuitionistic fragment augmented with classical axioms that state excluded middle at each type. The intuitionistic fragment is Curry-Howard isomorphic to Gentzen LK sequent calculus.

Table 1 shows the reduction rules of the calculus, according to [2]. The first two rules may form a critical pair. Consistently solving the critical pair by giving priority to one of the two rules leads to a call-by-value (respectively call-by-name) strategy, that this way are shown to be perfectly dual. The classical fragment of the calculus is not deterministic, since for a critical pair there may be no common reduct to form a diamond. However, the intuitionistic fragment is deterministic and it is a closed subset with respect to reduction. The $\overline{\lambda}\mu\tilde{\mu}$-calculus typing rules and its principal meta-theoretical properties can be found in [2].

3 Representing Alternatives in Proofs in the $\overline{\lambda}\mu\tilde{\mu}$-calculus

We are now interested in representing alternatives proofs and views in the $\overline{\lambda}\mu\tilde{\mu}$-calculus. Moreover, we want to avoid the addition to the calculus of new constructs and we also want to exploit a fragment as close as possible to the intuitionistic one. The latter requirement is necessary to preserve the good behavior of our rendering semantics that gives natural results only on that fragment.

The main idea underlying our encoding is that of seeing a view over a proof term just as reduced forms of the proof term (according to the reduction rules of the calculus). For each pair of alternative proofs in the calculus we have two possible set of views: one that picks the first view and one that picks the opposite one. Thus a proof term with an alternative must reduce non deterministically in two possible ways. This suggests that we must encode a pair of alternative proofs as a critical pair.

The $\overline{\lambda}\mu\tilde{\mu}$-calculus typing rules and the previous requirement suggest the following minimal encoding of a pair of alternative terms t_1 and t_2 that prove T:

$$\boxed{alt_r^T(t_1,t_2) := \mu\star : T.\langle \mu\_ : T.\langle t_1 || \star\rangle || \tilde{\mu}\_ : T.\langle t_2 || \star\rangle\rangle}$$

By duality we can also provide a continuation that embeds two alternative continuations E_1 and E_2 of type T. However, we will not need it in this paper and we consider as future work the study of fragments of the calculus that include it.

In the definition of $alt_r^T(t_1, t_2)$ we have used the notation $\mu_\_ : T$ ($\tilde{\mu}_\_ : T$) to remark that the bound term (continuation) will not occur in its scope. We also say that this occurrence of the binder is *affine*.

As requested, the term $alt_r^T(t_1, t_2)$ is subject to the following non deterministic reduction rules: $alt_r^T(t_1, t_2) \rhd \mu\star : T.\langle t_1 || \star\rangle$ and $alt_r^T(t_1, t_2) \rhd \mu\star : T.\langle t_2 || \star\rangle$. Notice that both right hand sides are μ-expanded forms respectively of t_1 and t_2 and thus they are extensionally equivalent to t_1 and t_2. Notice also that if t_1 and t_2 are both terms in the intuitionistic fragment then $alt_r^T(t_1, t_2)$ reduces only to terms in the intuitionistic fragment. Unfortunately, when the term is plugged into a command, it is also subject to the following reduction rule:

$$\langle alt_r^T(t_1, t_2) || E\rangle \rhd \langle \mu_\_ : T.\langle t_1 || E\rangle || \tilde{\mu}_\_ : T.\langle t_2 || E\rangle\rangle$$

The right hand side of the reduction rule is basically equivalent to the left hand side and it will enjoy all its interesting properties. However, the environment E is duplicated. According to our rendering semantics, the left hand side represents a large proof with two alternative subproofs. The right hand side represents two alternative large proofs that have duplicated parts. Thus we will be interested in preventing this form of reduction. Notice that we can easily syntactically detect the redexes we do not want to reduce. They are the redexes of the form:

$$\langle \mu\star : T.\langle \mu_\_ : T'.c_1 || \tilde{\mu}_\_ : T''.c_2\rangle || E\rangle$$

Finally, the reader can check the following typing derivation for our encoding according to the typing rules given in [2]:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash t_1 : T | \star : T; \Delta \qquad \overline{\Gamma | \star : T \vdash \star : T; \Delta}}{\langle t_1 || \star\rangle : \Gamma \vdash \star : T; \Delta}}{\Gamma \vdash \mu_\_ : T.\langle t_1 || \star\rangle : T | \star : T; \Delta} \qquad \dfrac{\dfrac{\Gamma \vdash t_2 : T | \star : T; \Delta \qquad \overline{\Gamma | \star : T \vdash \star : T; \Delta}}{\langle t_2 || \star\rangle : \Gamma \vdash \star : T; \Delta}}{\Gamma | \tilde{\mu}_\_ : T.\langle t_2 || \star\rangle : T \vdash \star : T; \Delta}}{\langle \mu_\_ : T.\langle t_1 || \star\rangle || \tilde{\mu}_\_ : T.\langle t_2 || \star\rangle\rangle : \Gamma \vdash \star : T; \Delta}}{\Gamma \vdash \mu\star : T.\langle \mu_\_ : T.\langle t_1 || \star\rangle || \tilde{\mu}_\_ : T.\langle t_2 || \star\rangle\rangle : T | \Delta}$$

The typing derivation shows a few peculiarities we are now going to analyze. First of all, when introducing affine binders we have not added the variables bound by an affine binder to the premises of the introduction rule. This is consistent with the original typing rules since the missing premise cannot play any role in the derivation because it is not referenced in the term.

Thanks to the previous observation, we notice that the continuation context Δ plays a passive role in the derivation, being simply propagated from the root to the leaves of the tree. As a consequence the derivation holds also when Δ is empty. In the latter case the continuation context has always exactly one declaration, and the two premises of the tree become

$$\Gamma \vdash t_1 : T | \star : T \quad \text{and} \quad \Gamma \vdash t_2 : T | \star : T$$

If t_1 and t_2 are terms in the intuitionistic fragment, then the continuation $\star$ declared in the context cannot occur in any of them. Thus it can be dropped from the typing judgment.

To summarize, if t_1 and t_2 are terms in the intuitionistic fragment, then the following "morally intuitionistic" derived rule applies for $alt_r^T(t_1, t_2)$:

$$\frac{\Gamma \vdash t_1 : T|\emptyset \qquad \Gamma \vdash t_2 : T|\emptyset}{\Gamma \vdash alt_r^T(t_1, t_2) : T|\emptyset}$$

By structural induction on the typing derivation, the same derivation holds for t_1 and t_2 in the intuitionistic fragment extended with $alt_r^-(-,-)$.

Following a similar line of reasoning, we will also admit the environment

$$\boxed{alt_l^T(t_1, t_2) := \tilde{\mu}x : T.\langle \mu\_ : T.\langle t_1 || \star\rangle || \tilde{\mu}\_ : T.\langle t_2 || \star\rangle\rangle}$$

and more generally commands of the form $\boxed{c ::= \langle v || E\rangle \mid \langle \mu\_ : T.c || \tilde{\mu}\_ : T.c\rangle}$

Excursus: the affine fragment of the $\overline{\lambda}\mu\tilde{\mu}$*-calculus* is the fragment where each $\tilde{\mu}$-binder is either affine or binds the continuation variable $\star$. The affine fragment is a superset of the one we adopt for encoding alternative proofs.

It is interesting to ask whether the whole affine fragment is a natural candidate for being a proof fragment. For the fragment we use this is a consequence of being essentially intuitionistic. Thus we can ask if the whole affine fragment is essentially intuitionistic or if it is inherently classical and does not admit a natural explanation of its proofs in pseudo-natural language.

According to the syntax of the $\overline{\lambda}\mu\tilde{\mu}$-calculus, an affine binder can occur only in two positions: as the first subterm of a command, possibly prefixed by lambda-abstractions ($\langle \lambda x_1 : T_1. \ldots \lambda x_n : T_n.\mu\_ : T.c || E\rangle$) — called a *spine* position — and as the first subterm of a "cons" environment, possibly prefixed by lambda-abstractions ($\lambda x_1 : T_1. \ldots \lambda x_n : T_n.\mu\_ : T.c \circ E$) — called an *argument* position. Since we are supposed to analyze an extension of an essentially intuitionistic fragment, we suppose that continuation variables range over the singleton $\{\star\}$. We show how to prove without assuming any axiom and in two different ways the classical statement $(A \Rightarrow C) \Rightarrow ((A \Rightarrow B) \Rightarrow C) \Rightarrow C$. As far as we know, both proofs do not admit a reasonable translation to natural language.

The first proof exploits an affine binder in argument position:

$$\lambda H_1 : A \Rightarrow C.\lambda H_2 : (A \Rightarrow B) \Rightarrow C.\mu\star : C.\langle H_2 || (\lambda x : A.\mu\_ : B.\langle H_1 || x \circ \star\rangle) \circ \star\rangle$$

The classical core of the proof is represented by the term $\lambda x : A.\mu\_ : B.\langle H_1 || x \circ \star\rangle$. The term has type $A \Rightarrow B$, but it does not conclude B under the hypothesis A. As soon as A is known by hypothesis, the hypothesis is used in conjunction with H_1 to jump to the outer proof and conclude C. In the intuitionistic fragment this would be prevented by the μ binder that starts the proof of B binding $\star$ and hiding the previous declaration of $\star$. In the affine fragment, however, the μ binder can bind no variable, without hiding $\star$. As far as we know, there is no structural way of providing a natural explanation of the proof term above in natural language.

The second proof, that exploits an affine binder in spine position, $\tilde{\mu}$-reduces to the first proof:

$$\begin{array}{l}\lambda H_1 : A \Rightarrow C.\lambda H_2 : (A \Rightarrow B) \Rightarrow C.\\ \quad \mu\star : C.\langle (\lambda x : A.\mu\_ : B.\langle H_1) || x \circ \star\rangle || \tilde{\mu}y : A \Rightarrow B.\langle H_2 || y \circ \star\rangle\rangle\end{array}$$

$$
\begin{array}{rl}
PROOF := & \mathsf{hyp}\ L{:}F;PROOF \\
\mid & \mathsf{alt}(PROOF_1 \mid PROOF_2) \\
\mid & \mathsf{derive}\ L : F\ JUST;PROOF \\
\mid & \mathsf{derive}\ \_ : F\ JUST \\
ARG := & L \mid (F\ PROOF)
\end{array}
\qquad
\begin{array}{rl}
JUST := & \mathsf{method}\ M(ARG_1 \ldots ARG_n)\ OPTEXP \\
\mid & \mathsf{plan}(ARG_1 \ldots ARG_n)\ (F\ PROOF) \\
\mid & \mathsf{sketch}(ARG_1 \ldots ARG_n) \\
OPTEXP := & \mathsf{nil} \mid (F\ PROOF)
\end{array}
$$

(where L are the labels of formulas, F the formulas, and M the references to methods.)
Table 2. Abstract Syntax for Formal OMDoc Proofs

We conclude that the affine fragment is too large to be directly useful as a proof format. The restriction obtained by allowing affine binders only if not prefixed by lambda-abstractions is essentially intuitionistic, but does not seem to add much expressive power with respect to the intuitionistic fragment extended with the affine binders that occur in $alt_l^-(-,-)$ only.

Natural language rendering of alternative proofs We equip the affine fragment considered to encode alternative proofs with the following *fully compositional* rendering semantics (according to [4]):

$$
\begin{array}{rl}
[\![\langle \mu_\_ : T.c_1 || \tilde{\mu}_\_ : T.c_2 \rangle]\!] = & \text{we provide two alternative proofs} \\
& \text{first proof:}\boxed{\hookrightarrow} \\
& \quad [\![c_1]\!] \\
& \text{alternative proof:}\boxed{\hookrightarrow} \\
& \quad [\![c_2]\!]
\end{array}
$$

The semantics is fully compositional since $[\![t_i]\!]$ occurs in output before $[\![t_j]\!]$ every time t_i occurs before t_j in the term. This is an important property since it allows one to translate a proof back and forth without any need to rearrange the subterms; in other words the translator can be implemented as a stream processor and a human being can easily understand a proof term running the translation in his head without need to take notes.

Views for alternative proofs, in the sense of [1], are produced from a proof (or a PDS) by picking just one alternative proof in each set of alternative choices. In the $\overline{\lambda}\mu\tilde{\mu}$-calculus we are considering, a view is obtained by reducing one of the two competing redexes of each critical pair in the proof term. This definition of a view is more informative than the corresponding one over PDSs and provides an effective guideline for extensions to more refined form of views.

4 Abstract Syntax for OMDoc Proofs

An extended OMDoc proof consists of hypothetical proof steps and proof steps that derive a new fact. These two proof steps correspond to the formal OMDoc proofs. In order to deal with alternative proof attempts, we add a third language construct marking the start of alternative proofs. The full grammar rules for proofs is given in Table 2. In Appendix A we present an adequate extension of the OMDoc document type definition to accommodate alternative proofs.

More specifically, hyp $L{:}F;PROOF$ denotes the start of a hypothetical proof, where scope of the hypothesis F with label L reaches until the end of the rest of the proof $PROOF$. Alternative subproofs are represented by alt($PROOF_1$ | $PROOF_2$). The derive proof steps are the most complex and of the general form

$$\textsf{derive } L : F \ JUST$$

Such a step states that we can derive the fact F by using the justification $JUST$. We allow three kinds of justifications $JUST$ for derive-steps: It can either be a sketch($ARG_1 \dots ARG_n$) in the sense proof sketches in [5] (called "gap steps" in [3]) or it is a reference to a method $M(ARG_1 \dots ARG_n)$ or it is the description of a subproof plan($ARG_1 \dots ARG_n$). In all cases the ARG_i are either premises or facts and associated proofs. Premises are referenced by their label L and a fact and its proof are represented by (F $PROOF$). If there is at least one fact and subproof, then this is a top-down proof step; otherwise it is a pure bottom-up proof step. A method can be calculus rule, but also a rule at some lower level of granularity which is associated with a proof at some higher-level of granularity. This accommodates the simultaneous representation of proofs for F at different levels of granularity that is necessary to encode the PDS [1]. In case the derive-step is the last step of a proof we require the labels L_i of the derived formulas to be undefined, which is indicated by _ in the grammar. Note that derive-steps are the only means to terminate a proof.

Further differences of the extended OMDoc proofs and the standard [3] are: (1) we consider only hypothesis and derivation steps that have *exactly one* formal formula (FMP) and (2) we do not allow for local declarations and definitions. The latter could easily be added and translated to $\overline{\lambda}\mu\tilde{\mu}$, but we omitted them for sake of simplicity. The first restriction is due to the fuzzy semantics of OMDoc proofs with several conclusions, that does not admit in the current state an explanation in $\overline{\lambda}\mu\tilde{\mu}$. Fixing the semantics of OMDoc will require also syntactic changes; we plan to do that in a future work, providing a correspondence with an already known extension of $\overline{\lambda}\mu\tilde{\mu}$ with primitive multiplicative conjunction.

The rendering semantics for extended OMDoc proofs is given in Table 3 where $[\![P]\!]$ denotes the rendering of the proof P. Due to the lack of space, we have omitted the explicit introduction of newlines. The rendering rules are straightforward: the difference between a derive-step being the final and thus concluding step of a proof or not is acknowledged by using the past tense *"... we proved..."* instead of the present tense *"... we prove..."* (for method application and proof descriptions) and *"... we show..."* (for proof sketches).

5 Encoding OMDoc in $\overline{\lambda}\mu\tilde{\mu}$ and the other way around

Tables 5 and 6 show forward and backward translations between OMDoc proofs in the full fragment and $\overline{\lambda}\mu\tilde{\mu}$-terms in the affine fragment of Table 4. The latter table also provides a rendering semantics for the fragment that is slightly more refined than the one given for the whole calculus.

$$\llbracket \mathsf{hyp}\ L{:}F;PROOF \rrbracket := \text{“}\textit{Assume } F\ (L)\llbracket PROOF \rrbracket\text{”}$$

$$\llbracket \mathsf{derive}\ L : F\ \mathsf{method}\ M(ARG_1,\ldots,ARG_k)\ OPTEXP; PROOF' \rrbracket$$
$$:= \text{“}\textit{By } M\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}\ \llbracket OPTEXP \rrbracket^{o}\ \textit{we prove } F\ (L); \llbracket PROOF' \rrbracket\text{”}$$

$$\llbracket \mathsf{derive}\ L : F\ \mathsf{plan}(ARG_1,\ldots,ARG_k)\ (F'\ PROOF); PROOF' \rrbracket$$
$$:= \text{“}\textit{We prove } F\ (L)\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}\ \llbracket (F'\ PROOF) \rrbracket^{o}; \llbracket PROOF' \rrbracket\text{”}$$

$$\llbracket \mathsf{derive}\ L : F\ \mathsf{sketch}(ARG_1,\ldots,ARG_k); PROOF' \rrbracket$$
$$:= \text{“}\textit{We show } F\ (L\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}; \llbracket PROOF' \rrbracket\text{”}$$

$$\llbracket \mathsf{derive}\ \_{:}F\ \mathsf{method}\ M(ARG_1,\ldots,ARG_k)\ OPTEXP \rrbracket$$
$$:= \text{“}\textit{By } M\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}\ \llbracket OPTEXP \rrbracket^{o}\ \textit{we proved } F; \boxed{\hookleftarrow}\text{”}$$

$$\llbracket \mathsf{derive}\ \_{:}F\ \mathsf{plan}(ARG_1,\ldots,ARG_k)\ \vdots\,(F'\ PROOF)\,\vdots \rrbracket$$
$$:= \text{“}\textit{We proved } F\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}\ \llbracket (F'\ PROOF) \rrbracket^{o}; \boxed{\hookleftarrow}\text{”}$$

$$\llbracket \mathsf{derive}\ \_{:}F\ \mathsf{sketch}(ARG_1,\ldots,ARG_k) \rrbracket$$
$$:= \text{“}\textit{We can obtain } F\ \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al}; \boxed{\hookleftarrow}\text{”}$$

$$\llbracket \mathsf{alt}(PROOF_1 \mid PROOF_2) \rrbracket := \text{“}\textit{Either } \boxed{\hookrightarrow} \llbracket PROOF_1 \rrbracket\ \textit{or } \boxed{\hookrightarrow} \llbracket PROOF_2 \rrbracket\text{”}$$

$$\llbracket () \rrbracket^{al} := \text{“”} \qquad \llbracket (ARG_1,\ldots,ARG_k) \rrbracket^{al} := \text{“}\textit{from } \llbracket ARG_1 \rrbracket^{a},\ldots,\ \textit{and } \llbracket ARG_k \rrbracket^{a}\text{”}$$

$$\llbracket \mathsf{nil} \rrbracket^{o} := \text{“”} \qquad \llbracket (F\ PROOF) \rrbracket^{a} := \text{“}F\ \textit{(proved by } \llbracket PROOF \rrbracket)\text{”}$$

$$\llbracket L \rrbracket^{a} := \text{“}L\text{”} \qquad \llbracket \vdots\,(F\ PROOF)\,\vdots \rrbracket^{o} := \text{“}(\textit{In detail: } \llbracket PROOF \rrbracket)\text{”}$$

Table 3. Rendering Semantics for Abstract OMDoc

Commands

$$c ::= \langle v' \| E \rangle \qquad \llbracket v' \rrbracket \llbracket E \rrbracket$$
$$\mid\ \langle \mu\_ : T.c_1 \ \|\tilde{\mu}\_ : T.c_2 \rangle \qquad \text{we provide two alternative proofs}\ \text{first proof:}\boxed{\hookrightarrow}\ \llbracket c_1 \rrbracket\ \text{alternative proof:}\boxed{\hookrightarrow}\ \llbracket c_2 \rrbracket$$

Environments

$$E ::= \star \qquad \boxed{\hookleftarrow}\ \text{done}$$
$$\mid\ \tilde{\mu}x : T.c \qquad \text{we proved } T\ (x)\ \llbracket c \rrbracket$$
$$\mid\ a \circ E \qquad \text{and } \llbracket a \rrbracket\ \llbracket E \rrbracket$$

ComplexArguments

$$a' ::= \mu\star : T.c \qquad \text{a proof of } T \boxed{\hookrightarrow}\ \llbracket c \rrbracket$$
$$\mid\ \lambda x : T.a' \qquad \text{under hypothesis } T\ (x)\ \llbracket a' \rrbracket$$

Arguments

$$a ::= x \qquad \text{by } x$$
$$\mid\ a' \qquad \llbracket a' \rrbracket$$

SpineTerms

$$v' ::= x \qquad \text{by } x$$
$$\mid\ ? \qquad \text{by a conjecture}$$
$$\mid\ v \qquad \textit{if } v = \mu\star : T.\langle v' \| \tilde{\mu}x : T.\langle x \| \star \rangle\rangle\ \textit{then}\ \text{by } x \text{ (that proves } T \text{ as follows } \boxed{\hookrightarrow}\ \llbracket \langle v' \| \star \rangle \rrbracket\ \textit{else}\ \text{by some proof (in detail } \boxed{\hookrightarrow}\ \llbracket \langle v' \| \star \rangle \rrbracket$$

Terms

$$v ::= \lambda x : T.v \qquad \text{suppose } T\ (x)\ \llbracket v \rrbracket$$
$$\mid\ \mu\star : T.c \qquad \text{we need to prove } T\ \boxed{\hookrightarrow} \llbracket c \rrbracket$$

Table 4. A $\overline{\lambda}\mu\tilde{\mu}$-fragment with its rendering semantics

$$
\begin{array}{l}
[\![\mathsf{hyp}\ L{:}F;PROOF]\!]^t_{F\to F'} = \lambda L : F.[\![PROOF]\!]^t_{F'}\\
[\![\mathsf{derive}\ \_{:}F\ JUST]\!]^t_F = \mu\star : F.\langle[\![JUST]\!]^j_1 || [\![JUST]\!]^j_2(\star)\rangle\\
[\![\mathsf{derive}\ L{:}F\ JUST;PROOF]\!]^t_{F'} = \mu\star : F'.\langle[\![JUST]\!]^j_1 || [\![JUST]\!]^j_2(\tilde{\mu}L : F.\underline{\langle[\![PROOF]\!]^t_{F'}||\star\rangle})\rangle\\
[\![\mathsf{alt}(PROOF_1 \mid PROOF_2)]\!]^t_F = alt^F_r([\![PROOF_1]\!]^t_F, [\![PROOF_2]\!]^t_F)\\
[\![\mathsf{sketch}(ARG_1 \dots ARG_n)]\!]^j_1 = ?\\
[\![\mathsf{sketch}(ARG_1 \dots ARG_n)]\!]^j_2(E) = [\![ARG_1]\!]^a \circ \dots \circ [\![ARG_n]\!]^a \circ E\\
[\![\mathsf{plan}(ARG_1 \dots ARG_n)\ \vdots\ (F\ PROOF)\ \vdots]\!]^j_1 = [\![PROOF]\!]^t_F\\
[\![\mathsf{plan}(ARG_1 \dots ARG_n)\ \vdots\ (F\ PROOF)\ \vdots]\!]^j_2(E) = [\![ARG_1]\!]^a \circ \dots \circ [\![ARG_n]\!]^a \circ E\\
[\![\mathsf{method}\ M(ARG_1 \dots ARG_n)\ \vdots\ (F\ PROOF)\ \vdots]\!]^j_1 = \mu\star : F.\langle[\![PROOF]\!]^t_F || \tilde{\mu}M : F.\langle M||\star\rangle\rangle\\
[\![\mathsf{method}\ M(ARG_1 \dots ARG_n)\ \vdots\ (F\ PROOF)\ \vdots]\!]^j_2(E) = [\![ARG_1]\!]^a \circ \dots \circ [\![ARG_n]\!]^a \circ E\\
[\![\mathsf{method}\ M(ARG_1 \dots ARG_n)\ \mathsf{nil}]\!]^j_1 = M\\
[\![\mathsf{method}\ M(ARG_1 \dots ARG_n)\ \mathsf{nil}]\!]^j_2(E) = [\![ARG_1]\!]^a \circ \dots \circ [\![ARG_n]\!]^a \circ E\\
[\![L]\!]^a = L\\
[\![(F\ PROOF)]\!]^a = [\![PROOF]\!]^t_F
\end{array}
$$

After the translation, underlined μ-redexes of the form $\underline{\langle\mu\star : T.c||\star\rangle}$ (also comprising the case $\mu\star : T.c \equiv alt^T_r(v_1, v_2)$) must be μ-reduced to c. Underlining of the remaining commands must be removed.

Table 5. OMDoc to $\overline{\lambda}\mu\tilde{\mu}$

$$
\begin{array}{ll}
 & [\![\lambda x : T.v]\!] = \mathsf{hyp}\ x{:}\ T;[\![v]\!]\\
 & [\![\mu\star : T.\langle v||v_1 \circ \dots \circ v_n \circ \star\rangle]\!] = \mathsf{derive}\ \_{:}T\ [\![v]\!]^m([\![v_1]\!]^a, \dots, [\![v_n]\!]^a)\\
 & [\![\mu\star : T.\langle v||v_1 \circ \dots \circ v_n \circ \tilde{\mu}x : T'.c\rangle]\!] = \mathsf{derive}\ x{:}T'\ [\![v]\!]^m([\![v_1]\!]^a, \dots, [\![v_n]\!]^a);[\![\mu\star : T.c]\!]\\
\dagger & [\![x]\!] = \text{ruled out}\\
 & [\![alt^T_r(v_1, v_2)]\!] = \mathsf{alt}([\![v_1]\!] \mid [\![v_2]\!])\\
 & [\![x]\!]^m(ARG_1, \dots, ARG_n) = \mathsf{method}\ x(ARG_1 \dots ARG_n)\\
 & [\![?]\!]^m(ARG_1, \dots, ARG_n) = \mathsf{sketch}(ARG_1, \dots, ARG_n)\\
\circ & [\![\mu\star : T.\langle v||\tilde{\mu}x : T.\langle x||\star\rangle\rangle]\!]^m(ARG_1, \dots, ARG_n) = \mathsf{method}\ x(ARG_1 \dots ARG_n)\ \vdots\ [\![v]\!]\ \vdots\\
 & [\![v]\!]^m(ARG_1, \dots, ARG_n) = \mathsf{plan}(ARG_1 \dots ARG_n)\ \vdots\ [\![v]\!]\ \vdots \quad \text{for } v \notin \{x, ?, \mu\star : T.\langle v||\tilde{\mu}x : T.\langle x||\star\rangle\rangle\}\\
 & [\![x]\!]^a = x\\
 & [\![\mu\star : T.c]\!]^a = (T\ [\![\mu\star : T.c]\!])\\
 & [\![alt^T_r(v_1, v_2)]\!]^a = (T\ [\![alt^T_r(v_1, v_2)]\!]\\
 & [\![\lambda x_1 : T_1. \dots \lambda x_n : T_n.\mu\star : T.c]\!]^a = (T_1 \Rightarrow \dots \Rightarrow T_n \Rightarrow T\ [\![\lambda x_1 : T_1. \dots \lambda x_n : T_n.\mu\star : T.c]\!])\\
 & [\![\lambda x_1 : T_1. \dots \lambda x_n : T_n.alt^T_r(v_1, v_2)]\!]^a = (T_1 \Rightarrow \dots \Rightarrow T_n \Rightarrow T\ [\![\lambda x_1 : T_1. \dots \lambda x_n : T_n.alt^T_r(v_1, v_2)]\!]\\
\dagger & [\![\lambda x_1 : T_1. \dots \lambda x_n : T_n.x]\!]^a = \text{ruled out}
\end{array}
$$

The rule marked with $\circ$ is necessary to make this translation inverse of the translation from OMDoc to $\overline{\lambda}\mu\tilde{\mu}$ (Table 5). The (error) rules marked with $\dagger$ are never applied when translating terms generated from OMDoc.

Table 6. $\overline{\lambda}\mu\tilde{\mu}$ to OMDoc

Notice that the grammar of the fragment can be simplified, for instance by identifying the $Term$ and $ComplexArg$ productions, that are kept distinct to the benefit of the presentation of the rendering semantics.

Indeed, the reader can check that the only intuitionistic terms of the calculus that are not in the fragment are x in spine position and $\lambda x_1 : T_1. \ldots \lambda x_n : T_n.x$ both in spine and argument position. In both cases a simple η-like μ-expansion can give an equivalent term in the fragment: x is expanded to $\mu\star : T.\langle x||\star\rangle$ and $\lambda x_1 : T_1. \ldots \lambda x_n : T_n.x$ to $\lambda x_1 : T_1. \ldots \lambda x_n : T_n.\mu\star : T.\langle x||\star\rangle$.

The new term "?" that can only occur as the first argument of a command is a linear placeholder for a missing term of the expected type.

By induction over OMDoc proof (respectively $\overline{\lambda}\mu\tilde{\mu}$-term) structure, it is possible to prove that the two translations behave as almost inverse functions. In particular $[\![-]\!]^t_F$ (in the first translation) is almost inverse of $[\![-]\!]$ (in the second one); $[\![-]\!]^j_1$ and $[\![-]\!]^j_2$ considered together are inverse of $[\![-]\!]^m$; the two $[\![-]\!]^a$ functions are inverse one of the other. The functions behave as inverse on every proof/term but

$$\mathsf{derive\ L : F}\ \mathit{JUST\ PROOF}_1; \mathsf{HYP\ L' : F'}; \mathit{PROOF}_2$$

that, translated to the $\overline{\lambda}\mu\tilde{\mu}$-calculus and back, becomes the richer term

$$\mathsf{derive\ L : F}\ \mathit{JUST\ PROOF}_1; \mathsf{derive}_{:}\mathsf{F''}\ \mathsf{plan()}\ (\mathsf{F''}\ \mathsf{HYP\ L' : F'}; \mathit{PROOF}_2)$$

that states explicitly what the hypothetical proof proves. The reader can check the rendering semantics associated to the two OMDoc proofs.

We illustrate the semantics provided to our abstract OMDoc proofs with the following abstract and partial proof of the irrationality of $\sqrt{12}$:

1. *The proof is by contradiction*
2. *We assume* $rat(\sqrt{12})$*;*
3. *We show there are* n, m*, such that* $int(n) \wedge int(m) \wedge \neg commondiv(n, m) \wedge \sqrt{12} = \frac{n}{m}$*;*
4. *By Lemma* $\sqrt{z} = \frac{x}{y} \Rightarrow z \times y^2 = x^2$ *we know* $12 \times m^2 = n^2$*;*
5. *We show* $commondiv(n, m)$*;*
6. *We have a contradiction.*

In this proof, the proof steps (3.) and (5.) are only descriptions of more complicated proofs which are made explicit in the encoding of this proof given in Fig. 1. Note further that the expansion of proof step (5.) contains alternative proofs for the shown statement[3] .

[3] Note that the second alternative of using the prime divisor 2 basically comes back to use the prime divisor 3. So it is a proof with detour, but it is a proof.

```
derive _ : ¬rat(√12)
 method ProofByContradiction ((rat(√12) ⇒ ⊥
     hyp L0 : rat(√12);
     derive L1 : int(n) ∧ int(m) ∧ ¬commondiv(n,m) ∧ √12 = n/m
      plan(L0)  (rat(√12) ⇒ int(n) ∧ int(m) ∧ ¬commondiv(n,m) ∧ √12 = n/m
                hyp L10 : rat(√12); derive L11 : ∃y:int, z:int.√12 = y/z ∧ ¬commondiv(y,z)
                                       method ApplyLemma(Rat-Criterion, L10) ;
                derive _ : int(n) ∧ int(m) ∧ ¬commondiv(n,m) ∧ √12 = n/m
                    method decomposition(L11))
     derive L2 : 12 × m^2 = n^2 method ApplyLemma(√z = x/y ⇒ z × y^2 = x^2,L1) ;
     derive L3 : commondiv(n,m)
      plan(L2)  (12 × m^2 = n^2 ⇒ commondiv(n,m)
                hyp L30 : 12 × m^2 = n^2
                alt ( derive L31 : div(n,3) ∧ div(m,3)
                        method and-I (div(n,3) ... ) (div(m,3) ... ); ...
                    | derive L34 : div(n,2) ∧ div(m,2)
                        method and-I (div(n,2) ... ) (div(m,2) ... ); ...);
     derive _ : ⊥ method Contradiction(L1, L3)))
```

Fig. 1. Part of a Proof of the irrationality of $\sqrt{12}$

More specifically, we illustrate the semantics by showing (1) the rendering of that proof using the rendering semantics from Table 3, (2) the $\overline{\lambda}\mu\tilde{\mu}$ resulting from the translation of that proof and (3) the rendering of that $\overline{\lambda}\mu\tilde{\mu}$-term using the $\overline{\lambda}\mu\tilde{\mu}$-rendering semantics from Table 4.

The rendering of the OMDoc proof from Fig. 1 is the following:

Proof: *By* ProofByContradiction *from* $rat(\sqrt{12}) \Rightarrow \bot$
(proved by: Assume $rat(\sqrt{12})$ *(*L_0*)*
We prove $int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}$ *(*L_1*) from* L_0
(In details: Assume $rat(\sqrt{12})$ *(*L_{10}*) By* ApplyLemma *from* Rat-Criterion *and* L_{10} *we prove* $\exists y{:}int, z{:}int.\sqrt{12} = \frac{y}{z} \wedge \neg commondiv(y,z)$ *(*L_{11}*)*
By decomposition *from* L_{11} *we proved* $int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}$*);*
By ApplyLemma *from* $\sqrt{z} = \frac{x}{y} \Rightarrow z \times y^2 = x^2$ *and* L_1 *we prove* $12 \times m^2 = n^2$ *(*L_2*);*
We prove $commondiv(n,m)$ *(*L_3*) from* L_2
(In details:
Assume $12 \times m^2 = n^2$ *(*L_{30}*)*
Either by and-I *from* $div(n,3)$ *(proved by* $[\![\ldots]\!]$*) and* $div(m,3)$ *(proved by* $[\![\ldots]\!]$*) we proved* $div(n,3) \wedge div(m,3)$ *(*L_{31}*); ...*
Or by and-I *from* $div(n,2)$ *(proved by* $[\![\ldots]\!]$*) and* $div(m,2)$ *(proved by* $[\![\ldots]\!]$*) we proved* $div(n,2) \wedge div(m,2)$ *(*L_{34}*); ...)*
By Contradiction *from* L_1 *and* L_3 *we proved* $\bot$*)*
we proved $\neg rat(\sqrt{12})$

The $\overline{\lambda}\mu\tilde{\mu}$-term obtained by translation using the rules Table 5 after μ-reduction of the underlined μ-redexes of the form $\langle \mu\star : T.\langle \mu_{\_} : T'.c_1 || \tilde{\mu}_{\_} : T''.c_2\rangle || E\rangle$ is shown in Fig. 2.

$\mu\star : \neg rat\sqrt{12}.$
$\langle$ProofByContradiction
$\quad \| \lambda L_0 : rat(\sqrt{12}).$
$\quad\quad \mu\star : \bot.\langle\ \mu\star : rat(\sqrt{12}) \Rightarrow int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}.$
$\quad\quad\quad \lambda L_{10} : rat(\sqrt{12}).$
$\quad\quad\quad\quad \langle$ApplyLemma $\|$
$\quad\quad\quad\quad\quad$ Rat-Criterion $\circ L_{10} \circ \tilde{\mu} L_{11} : \exists y{:}int, z{:}int.\, \sqrt{12} = \frac{y}{z} \wedge \neg commondiv(y,z).$
$\quad\quad\quad\quad\quad\quad \langle$decomposition $\| L_{11} \circ \star\rangle\rangle$
$\quad\quad\quad \| L_0 \circ \tilde{\mu} L_1 : int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}.$
$\quad\quad\quad\quad \langle$ApplyLemma $\|$
$\quad\quad\quad\quad\quad [\![\sqrt{z} = \frac{x}{y} \Rightarrow z \times y^2 = x^2]\!] \circ L_1$
$\quad\quad\quad\quad\quad\quad \circ\tilde{\mu} L_2 : 12 \times m^2 = n^2.$
$\quad\quad\quad\quad\quad\quad\quad \langle\ \lambda L_{30} : 12 \times m^2 = n^2.$
$\quad\quad\quad\quad\quad\quad\quad\quad alt_r(\mu\star : commondiv(n,m).$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad \langle$and-I $\| [\![(div(n,3)\ldots)]\!] \circ [\![(div(m,3)\ldots)]\!] \circ$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \tilde{\mu} L_{31} : div(n,3) \wedge div(m,3).\langle[\![\ldots]\!], \| \star\rangle\rangle,$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad \mu\star : commondiv(n,m).$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad \langle$and-I $\| [\![(div(n,2)\ldots)]\!] \circ [\![(div(m,2)\ldots)]\!] \circ$
$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad \tilde{\mu} L_{34} : div(n,2) \wedge div(m,2).\langle[\![\ldots]\!], \| \star\rangle\rangle)$
$\quad\quad\quad\quad\quad\quad\quad \| L_2 \circ \tilde{\mu} L_3 : commondiv(n,m).$
$\quad\quad\quad\quad\quad\quad\quad\quad \langle$Contradiction $\| L_1 \circ L_3 \circ \star\rangle\rangle \circ \star\rangle\rangle\rangle$

Fig. 2. $\overline{\lambda}\mu\tilde{\mu}$-Proof obtained by translation and after reduction of μ-redexes.

The rendering of the $\overline{\lambda}\mu\tilde{\mu}$-proof from Fig. 2 according to Table 4 yields:

Proof: *we need to prove* $\neg rat(\sqrt{12})$
by ProofByContradiction and under the hypothesis $rat(\sqrt{12})$ *(L_0) a proof of* $\bot$
by some proof
(in detail: we need to prove $rat(\sqrt{12}) \Rightarrow int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}$*: Suppose* $rat(\sqrt{12})$ *(L_{10}); By ApplyLemma and Rat-Criterion and L_0 we proved* $\exists y{:}int, z{:}int.\, \sqrt{12} = \frac{y}{z} \wedge \neg commondiv(y,z)$ *(L_{11}). By decomposition and L_{11}. Done)*
and L_0 we proved $int(n) \wedge int(m) \wedge \neg commondiv(n,m) \wedge \sqrt{12} = \frac{n}{m}$*;*
By ApplyLemma and $\sqrt{z} = \frac{x}{y} \Rightarrow z \times y^2 = x^2$ *and L_1 we proved* $12 \times m^2 = n^2$ *(L_2);*
By some proof
(in detail: Suppose $12 \times m^2 = n^2$ *(L_{30}); we provide two alternative proofs:*
First proof: we need to prove $commondiv(n,m)$*: By and-I and* $[\![(div(n,3)\ldots)]\!]$ $[\![(div(m,3)\ldots)]\!]$ *we proved* $div(n,3) \wedge div(m,3)$*;* $[\![\ldots]\!]$ *Done.*
Second proof: we need to prove $commondiv(n,m)$*: By and-I and* $[\![(div(n,2)\ldots)]\!]$ $[\![(div(m,2)\ldots)]\!]$ *we proved* $div(n,2) \wedge div(m,2)$*;* $[\![\ldots]\!]$ *Done.)*
and L_2 we have proved $commondiv(n,m)$ *(L_3);*
By Contradiction and L_1 and L_3 done.
Done.

The informative content of the natural language explanation obtained from OMDoc and that obtained from the corresponding $\overline{\lambda}\mu\tilde{\mu}$-calculus encoding are clearly almost equivalent. The two main differences are omission of repetitions of the thesis: (1) in the two alternative proofs the local thesis is restated in the $\overline{\lambda}\mu\tilde{\mu}$-calculus, but not in OMDoc; (2) at the end of the first expanded proof OMDoc states again what has been proved while the $\overline{\lambda}\mu\tilde{\mu}$-calculus does not. It would certainly be possible as a future work to change one or both renderings to obtain two syntactically closer texts. However, the interest would be limited, since we are already convinced that the informative content is equivalent and since neither of the two is really more readable or natural than the other.

6 Conclusion

In this paper we have continued the investigation of the $\overline{\lambda}\mu\tilde{\mu}$-calculus as a proof format, including the representation of alternative proofs and proofs at different level of details. Alternative proofs can be easily accommodated in a non deterministic fragment of the calculus that remains essentially intuitionistic, admitting proof explanation in a pseudo-natural language.

We have also demonstrated how it is possible to establish a tight correspondence between OMDoc and the $\overline{\lambda}\mu\tilde{\mu}$-calculus, that imposes a clear understanding of OMDoc proofs (and, indirectly, for the corresponding PDSs) in terms of proofs in a given logic. It is now possible to speak, for instance, of cut elimination for OMDoc, considering the operation inherited by the formal correspondence.

The pseudo natural language generation considered in this paper is to be understood as a textual representation of the proof that allows to understand it in all its details. It is not meant to be a *nice* or *natural* description of the proof – that we leave to experts in linguistics. However, it is important to provide it to further constrain the translation between the two proof formats: only a translation that essentially preserves the two independently given rendering semantics is acceptable, pruning out irrelevant embeddings between mathematically unrelated formats that can represent anything (like Lisp S-expressions or plain XML).

Finally, a few difficulties we have faced in establishing the correspondence could be understood as flaws in the OMDoc recommendation and could guide the future development of the language. For instance, the fact that hypothetical proof steps that follow derive steps when translated to $\overline{\lambda}\mu\tilde{\mu}$-calculus and read back are enriched with the statement the hypothetical step is proving. Concretely, the abstract syntax for OMDoc proofs we have considered already represents an extension of OMDoc that clarifies the role of proofs at different levels of detail and that adds alternative proofs.

As a future work we plan to continue the study of the correspondence between the two languages and their natural language renderings, in order to improve OMDoc and to unveil other remarkable properties of bigger and bigger fragments of the $\overline{\lambda}\mu\tilde{\mu}$-calculus used as a proof format.

References

1. S. Autexier, C. Benzmüller, D. Dietrich, A. Meier, C. Wirth. "A Generic Modular Data Structure for Proof Attempts Alternating on Ideas and Granularity". In Fourth International Conference on Mathematical Knowledge Management (MKM2005), Lecture Notes in Artificial Intelligence (LNAI), Vol. 3863, 2006.
2. P. Curien, H. Herbelin. "The duality of computation". In Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP'00), ACM, SIGPLAN Notices 35(9), ISBN:1-58113-2-2-6, 233–243, 2000.
3. M. Kohlhase. *OMDoc: An Open Markup Format for Mathematical Documents (Version 1.2).*
4. C. Sacerdoti Coen. "Explanation in Natural language of $\overline{\lambda}\mu\tilde{\mu}$-terms". In Fourth International Conference on Mathematical Knowledge Management (MKM2005), Lecture Notes in Artificial Intelligence (LNAI), Vol. 3863, 2006.
5. F. Wiedijk. "Formal Proof Sketches". In S. Berardi, M. Coppo and F. Damiani eds., Types for Proofs and Programs: Third International Workshop, TYPES 2003, LNCS 3085, 378–393, 2004.

A DTD Extension for OMDoc with Alternative Proofs

```
<!--
   An XML DTD for Open Mathematical documents (OMDoc 1.2) Module PF
      SYSTEM http://www.mathweb.org/omdoc/dtd/omdocpf.mod
      PUBLIC -//OMDoc//ELEMENTS OMDoc PF V1.2//EN
   See the documentation and examples at http://www.mathweb.org/omdoc
   (c) 1999-2003 Michael Kohlhase, released under the GNU Public License (GPL)

   -- Added element <alt> for alternative proofs
      (Serge Autexier & Claudio Sacerdoti-Coen (Mai 2006))

-->

<!-- qnames for omdoc statements -->
<!ENTITY % omdocpf.metacomment.qname "%omdoc.pfx;metacomment">
<!ENTITY % omdocpf.derive.qname "%omdoc.pfx;derive">
<!ENTITY % omdocpf.hypothesis.qname "%omdoc.pfx;hypothesis">
<!ENTITY % omdocpf.method.qname "%omdoc.pfx;method">
<!ENTITY % omdocpf.premise.qname "%omdoc.pfx;premise">
<!ENTITY % omdocpf.alternatives.qname "%omdoc.pfx;alt">

<!ELEMENT %omdocpf.proof.qname;
          (%omdocdoc.meta.content;
           (%ss;|%omdocmtxt.omtext.qname;
                |%omdocst.symbol.qname;
                |%omdocst.definition.qname;
                |%omdocpf.derive.qname;
                |%omdocpf.hypothesis.qname;
           )*
           (%omdocpf.alternatives.qname;)?
```

```
          )>

<!ATTLIST %omdocpf.proof.qname;
          %omdoc.common.attribs;
          %omdoc.toplevel.attribs;
          %fori.attrib;>

<!ELEMENT %omdocpf.proofobject.qname;
          (%omdocdoc.meta.content;(%omdocmobj.class;))>
<!ATTLIST %omdocpf.proofobject.qname;
          %omdoc.common.attribs;
          %omdoc.toplevel.attribs;
          %fori.attrib;>

<!ELEMENT %omdocpf.derive.qname;
          (%omdocmtxt.MCF.content;,(%ss;|%omdocpf.method.qname;)?)>
<!ATTLIST %omdocpf.derive.qname;
          %omdoc.common.attribs;
          type CDATA #IMPLIED
          %id.attrib;>

<!ELEMENT %omdocpf.hypothesis.qname; (%omdocmtxt.MCFS.content;)>
<!ATTLIST %omdocpf.hypothesis.qname;
          %omdoc.common.attribs;
          %id.attrib;
          inductive (yes|no) #IMPLIED>

<!ELEMENT %omdocpf.alternatives.qname;
          (%omdocpf.proof.qname;
           (%omdocpf.proof.qname;)+
          )>

<!ATTLIST %omdocpf.alternatives.qname;
          %omdoc.common.attribs;
          %id.attrib;
          >

<!ELEMENT %omdocpf.method.qname;
                  (%omdocmobj.class;|%omdocpf.premise.qname;
                  |%omdocpf.proof.qname;|%omdocpf.proofobject.qname;)*>
<!ATTLIST %omdocpf.method.qname; %omdoc.common.attribs; %xrefi.attrib;>
<!-- 'xref' is a pointer to the element defining the method -->

<!ELEMENT %omdocpf.premise.qname; EMPTY>
<!ATTLIST %omdocpf.premise.qname; %omdoc.common.attribs;
          %xref.attrib; rank CDATA "0">
<!-- The rank of a premise specifies its importance in the
     inference rule. Rank 0 (the default) is a real premise,
     whereas positive rank signifies sideconditions of
     varying degree. -->
```