MMT Objects

Florian Rabe

Jacobs University, Bremen, Germany

OpenMath 2014

Overview

- Major OpenMath-based experiment/system
- MMT
  - Universal representation language for formal logical content
    inspired by OpenMath, OMDoc
  - Implementation with generic support for logical and knowledge management functionality
    e.g., module system, type reconstruction; presentation, editing
- Object layer uses OpenMath as primary data structure

Point of This Talk

- Describe differences between MMT objects and OPENMATH objects
- Provide additional information for further development of OPENMATH
- Not a
  - position paper
  - standard enhancement proposal

  MMT's deviations may or may not be good for OPENMATH

Grammars

c: reference to symbol/constant (OMS)
x: reference to variable (OMV)

OpenMath

$$
\begin{array}{llcl}
\text{objects} & O & ::= & \mathcal{I}(i) \mid \mathcal{F}(f) \mid \mathcal{S}(s) \mid \mathcal{BA}(b) \\
 & & \mid & c \mid x \\
 & & \mid & \mathcal{A}(O, O^*) \mid \mathcal{ATT}(O; KV^*) \\
 & & \mid & \mathcal{B}(O; \mathcal{ATT}(x; KV^*)^*; O) \mid \mathcal{E}(c; O^*) \\
\text{(key-values lists)} & KV & ::= & c \mapsto O
\end{array}
$$

Mmt

$$
\begin{array}{llcl}
\text{objects} & E & ::= & \mathcal{L}^c(s) \\
 & & \mid & c \mid x \\
 & & \mid & c(\gamma; \Gamma; E^*) \\
\text{contexts} & \Gamma & ::= & (x[: E][= E])^* \\
\text{substitutions} & \gamma & ::= & (x = E)^*
\end{array}
$$

Literals

OpenMath

- 4 fixed literal types: integers, float, string, byte array
- concrete syntax fixed by standard
- side note: OpenMath standard CDs define no operations on strings or byte arrays

Mmt literals $\mathcal{L}^c(s)$

- extensible set of literal types   like extensible set of symbols
- no individual literal types built-in
- c is symbol whose documentation defines
  - syntax (string encoding)
  - semantics (valid values and their meaning)

  of string s, which represents the literal value

Attributions

OpenMath

- attributed variables in particular needed for type attributions
- semantically attributed objects does anybody use this?
- ignorable attributions

Mmt: no attributions

- contexts declare variables $x[: E][= E]$
  effectively 2 built-in attribution keys

$$\mathcal{ATT}(x; [\mathtt{type} \mapsto T], [\mathtt{def} \mapsto D]) \quad \simeq \quad x[: T][= D]$$

- ignorable attributions as extra-linguistic metadata
  somewhat similar to HTML + RDFa

Errors

OpenMath

- Explicit error objects

Mmt: no errors

- error objects recovered as special case of application objects

Complex Objects

OpenMath

- 4 constructions: attribution of key-value list, error, application, binding
- Note:
  - attribution and binding are purely structural
  - error implies semantic properties
  - application is in between

is function application semantics implied or not?

Mmt

- single construction $c(\gamma; \Gamma; \vec{E})$
- purely structural
  - named children γ
  - bound variables Γ
  - unnamed children (in scope of bound variables)
- each construction labeled with symbol c
- semantics of $c(\gamma; \Gamma; \vec{E})$ defined solely by semantics of c

Complex Objects (2)

OpenMath-Mmt correspondence $O \quad \simeq \quad E$
If

$$O_i \quad \simeq \quad E_i \quad \text{and} \quad V_j \quad \simeq \quad X_j,$$

then for applications:

$$\mathcal{A}(c, O_1, \ldots, O_n) \quad \simeq \quad c(\cdot; \cdot; E_1, \ldots, E_n)$$

bindings:

$$\mathcal{B}(c; V_1, \ldots, V_m; O_1) \quad \simeq \quad c(\cdot; X_1, \ldots, X_n; E_1)$$

errors:

$$\mathcal{E}(c; O_1, \ldots, O_n) \quad \simeq \quad c(\cdot; \cdot; E_1, \ldots, E_n)$$

Complex Objects (3)

- What does γ do in $c(\gamma; \Gamma; \vec{E})$?
- Generalization beyond application and binding objects
- Substitution γ used for
  - named arguments in function application
  - records
  - list of cases in pattern-match

Conclusion

- MMT grammar uses only 4 productions
  - constants
  - variables
  - literals
  - complex objects
- OPENMATH uses 10 productions
  - 4 kinds of literals
  - 4 kinds of complex objects
- MMT loses some expressivity, especially for applications
- But gained simplification crucial in MMT implementation