Towards a

MIZAR Mathematical Library

in OMDOC format

joint work with

Grzegorz Bancerek

MICHAEL KOHLHASE

School of Computer Science

Carnegie Mellon University

http://www.cs.cmu.edu/˜kohlhase

MBASE, a Knowledge Base of Math. Theories

- This has been attempted before! (Principia Math., Bourbaki,. . .)
- This time stress the infrastructure aspect (Open Source Model)
  - enable easy and powerful browsing (personalization, MATHML)
  - high-level (semantic) search (commutativity: $X(Y,Z) = X(Z,Y)$)
  - distributed Internet support: (e.g. local working KB vs. archive KB)
  - version management and concurrent access (like CVS for cooperation)
  - offer added-value inference services (enlist MATHWEB)
  - large-scale structure for navigation & reuse (theory graph, inheritance)
- MBASE as a MATHWEB component. (not only for human consumption)
  - situated vs. stateless communication of mathematical services

The MIZAR Mathematical Library

- Large body of machine-checked mathematical knowledge.
- First-order formalization (based on set theory)
- Under development for about 30 years.
- Numbers: ∼ 750 articles, ∼ 33000 theorems, ∼ 6200 definitions
- Syntax: Human-oriented text representation (∼ 60 MB)

```
for B1 being  set holds
  B1 is finite
   iff
  ex B2 being Relation-like Function-like set
     st (rng B2 = B1 & dom B2 in omega)
```

- What a wonderful Corpus to test MBASE on. (need OMDOC format)

OMDoc in a Nutshell (three levels of modeling)

Formula level: OpenMath/C-MathML

- Objects as logical formulae
- semantics by ref. to theory level

```
<OMA>
 <OMS cd="arith1" name="plus"/>
 <OMS cd="nat" name="zero"/>
 <OMV name="N"/>
</OMA>
```

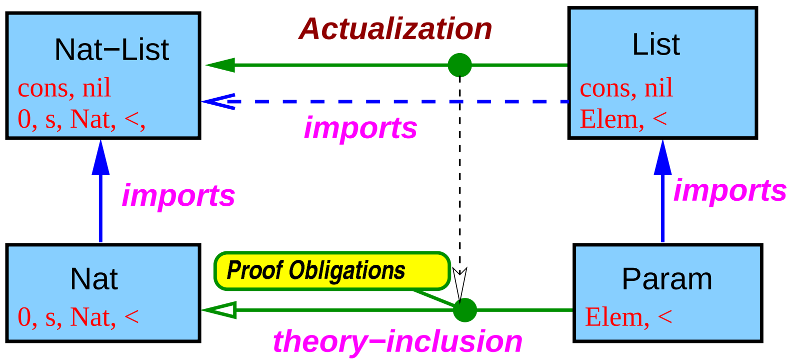
Statement level:

- Definition, Theorem, Proof, Example
- semantics explicit forms and refs.

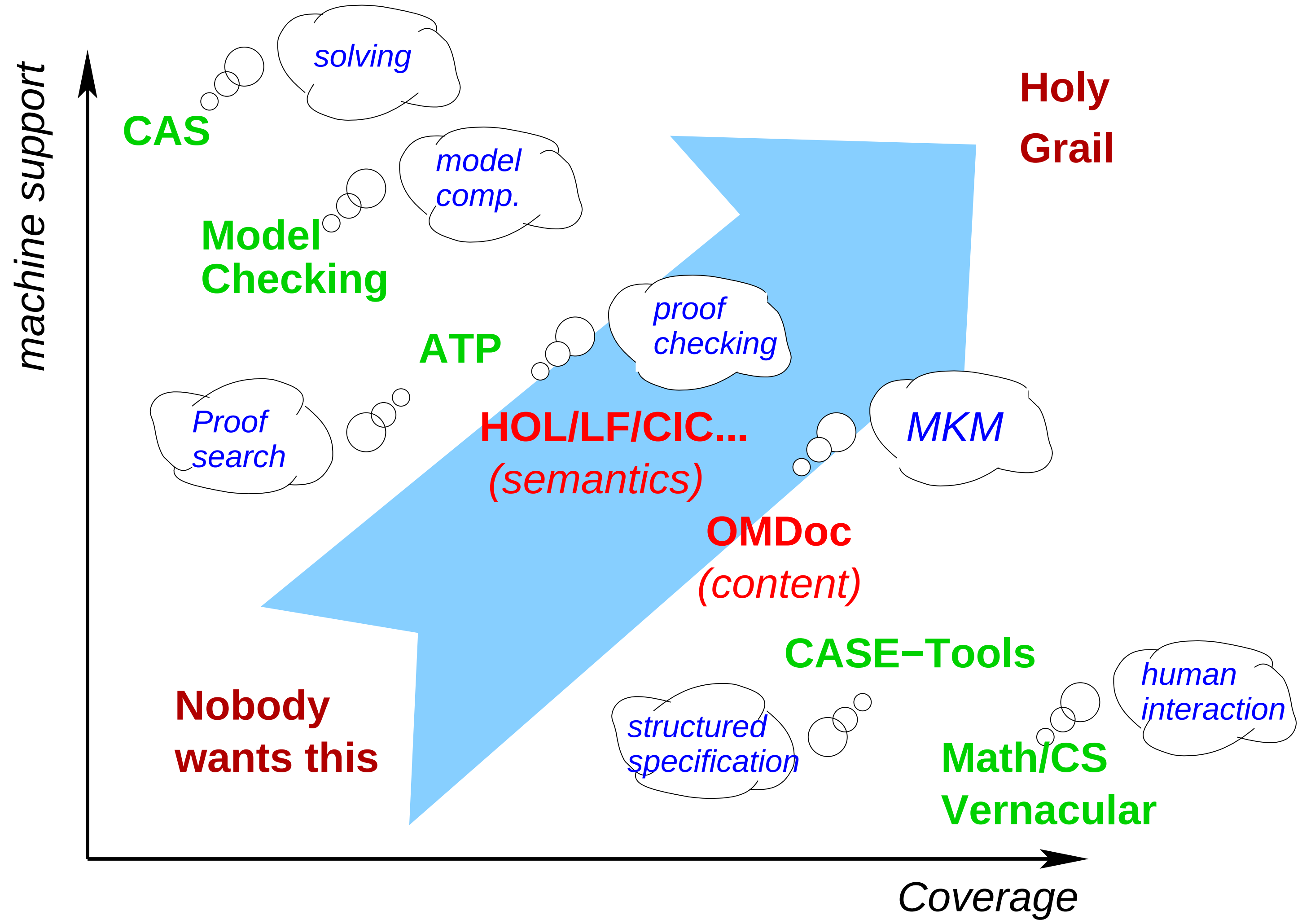
```
<defn for="plus" type="rec">
 <CMP>rec. eq. for plus</CMP>
 <FMP>X+0 = 0</FMP>
 <FMP>X+s(Y) = s(X+Y)</FMP>
</defn>
```

Theory level: Development Graph

- inheritance via symbol-mapping
- theory-inclusion by proof-obligations
- local (one-step) vs. global links



Situating Content Markup: Knowledge Management



Translating MIZAR: a brief history

[JFM] HTML/LaTeX presentations for the Journal of Formalized Mathematics (informal: human-oriented)

[Various] Hand translations for benchmark examples (by various people)

[Bylinski & Dahn 1997/8] Translations into PROLOG syntax (unfinished)

[Urban 2003] Translations into FOL for ATP. (Verify MIZAR with ATP)

- standard relativization for types (standard FOL, but loses structure)
- No MIZAR schemes and Fränkel operators. (Higher-Order)

[Bancerek 2002] MMLQUERY data (Information Retrieval for MKM)

- structure-preserving (no relativization)
- no proofs (under construction) (deemed irrelevant for IR)

General Issues in the Translation

- Pattern-level vs. Constructor-level ([Urban ’03])
  - MIZAR allows synonyms for and overloading of symbols
  - Pattern-level: Human-oriented rich language
  - Constructor-level: Unique machine representation (Bancerek: “strict MIZAR”)

The Translation MMLQuery Data to OMDoc

Idea	Information Retrieval + Arrangement ⇝ Translation
Concretely	MMLQuery + OMDoc templates ⇝ OMDoc

- Sample Template:

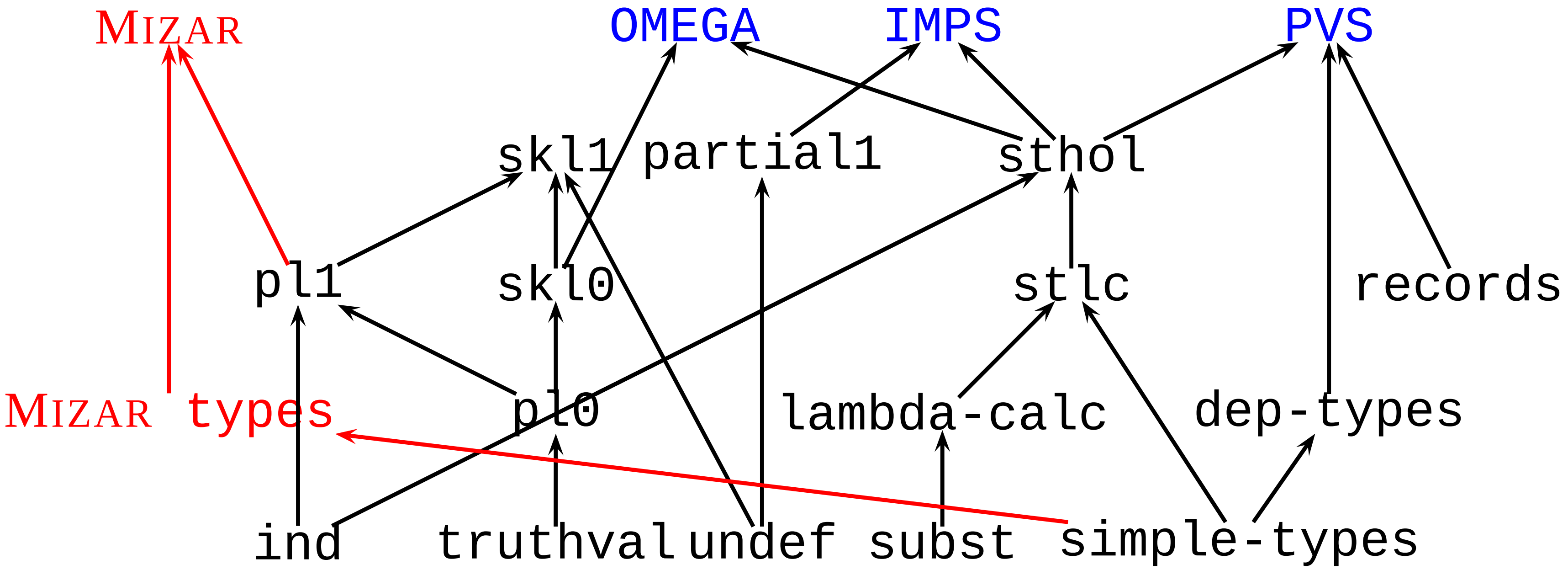
```
<symbol id="<mmlq type="kind"/><mmlq type="number"/>">
  <commonname><mmlq type="value" operation="[firstnot|symbol]"/></commonname
  <type system="mizar">
    <OMOBJ><mmlq type="fmp"/></OMOBJ>
  </type>
</symbol>
```

- Sample Result:

```
<symbol id="pred1">
  <commonname>=</commonname>
  <type system="mizar">
    <OMOBJ>
      <OMA><OMS cd="mizar" name="pred-type"/>
        <OMS cd="hidden" name="mode1"/><OMS cd="hidden" name="mode1"/>
      </OMA>
    </OMOBJ>
  </type>
</symbol>
```

A standardized Hierarchy of logical languages

- Idea: Provide a standardized, well-documented set of "names" for logical languages



- This hierarchy is based on literal inclusion (can we do better?)
- MBASE: Conservative Extension Principle with Logic Morphisms (Extend the Hierarchy with a level of proofs.)

The “MIZAR-types” theory for the MIZAR language

`type` The type attribute. It relates an object to its type in an attribuition.

`func-type` The function type constructor it takes n arguments. The last child is the range type and the first n-1 ones are the domain types.

`pred-type` The predicate type constructor it takes n arguments that are the domain types of the relation.

`type-expression` The type expression constructor. The first argument is a radix type and the rest is a list of adjectives that modify it.

`adjective-cluster` The adjective composition operator. It makes an adjective cluster out of a sequence of simpler ones.

The “MIZAR” theory for the MIZAR language (I)

`type-expression-nonempty` The operator that expresses the fact that a type is legal, since it is non-empty.

`adjective-inclusion` The operator that expresses the fact that an adjective includes another on a radix type. The first argument is the radix type, the next two are adjective clusters, the first cluster entails the second one on the radix.

`is` The ”is of type” operator in mizar. It expresses that its first argument has the type that is expressed as its second argument.

`possesses` The operator the expresses that a term possesses certain properties. Its first argument is the term and the rest are adjectives that express the properties.

`adjective-holds` The operator that expresses the fact that an adjective always holds on a radix type. The first argument is the radix type, the second one is an adjective cluster.

The “MIZAR” theory for the MIZAR language (II)

`fraenkel-bind` The finding part of the Fraenkel Operator

`fraenkel-set` The set part of the Fraenkel Operator.

Status of the Translation

- structure preserving translation of all statement-level information (except proofs)
  - produces valid OMDoc, transforms to TEX, MathML...
- work in progress (needed for completeness, full confidence)
  - theory hierarchy, notation definitions, proofs
  - back-translation: OMDoc $\rightsquigarrow$ strict Mizar $\rightsquigarrow$ full Mizar
- Future work (this is what motivates the work)
  - integrate Mizar theory with existing ones via “logic morphisms” (defines Mizar language, ensures translation)
  - re-grouping of knowledge base by topology/clustering
  - Extended pattern search/MMLQuery in MBase.