MP, as Input to OpenMath

Simon Gray, Norbert Kajler

September 1, 1995

Clear Focus

MP focuses on the efficient exchange of math expressions.

- 8 families of types:
  $\rightarrow$ Basic types: integer, real, identifier, constant, string, raw, meta
  $\rightarrow$ Composite types: operator
- Annotations: required, supplemental, user- or MP- defined
- *Meta-information* flags **require** the sender to provide some minimal semantics information
- Dictionaries

$\Rightarrow$ enough to efficiently exchange mathematical expressions with shared meaning

MP Status

- MP 1.0 defines a single, binary encoding
- MP 1.0 comes with a reference implementation in C
- Additional APIs for other languages built on top of the C library through foreign function call mechanism:
  → Common-Lisp API under development
- Possibility to write alternative APIs based on the encoding specification

Syntax vs Semantics

- Separate syntax and semantics issues:
  → Possibility to use the syntactic-level only
  → Message can always be decoded at the syntactic-level

- Enforce minimal semantic information to be attached to each symbolic node:
  → Does the symbol carry a specific meaning ?
  → If so, is the meaning available to the receiver ?
  → If so, is it MP-defined ?

- Dictionaries allow attaching additional semantic information.

Binary Encoding

- MP encoding is binary (and binary **only**)
- Binary → ASCII → Binary is a trivial conversion
- Binary is far more efficient than ASCII for numeric data (efficiency is essential for some applications)
- Binary is **not** inefficient for symbolic data
- Binary provides an efficient way to pack extra information through bit flags.

MP 1.0 Additions

- Make the common case fast via "common" types:
  $\rightarrow$ new types: common integer, operator, constant, and identifier
- Other new types: boolean, constant
- Common dictionaries for common constant and operator
  $\rightarrow$ lookups can be done very quickly
- Meta-information flags

New Experimentations

- On-going projects connecting:
  → symbolic - symbolic systems
  → symbolic - numeric systems
  → symbolic - graphing systems
  → modules in a PDE solving environment (bus-based)
- Ports to parallel machines (Cray, T3D) through local supercomputing center

Conclusion

- Goals are straightforward, unambiguous, realizable
- Binary encoding is efficient and practical
- Dictionaries provide basis for shared meaning
- Flags provide simple, efficient way to guarantee essential information is provided
- Continued experimentation provides refinements (as nothing can be perfect the first time)

$\Rightarrow$ start simple, learn from what you do, grow