Unifying Math Ontologies: A Tale of Two Standards

Differentiating between analysis and algebra

James Davenport & Michael Kohlhase

University of Bath (visiting Waterloo) &
Jacobs Universität Bremen

12 July 2009

Thanks to referees, and many in OpenMath and MathML

The views expressed, though, are our own

A thought

Whenever anyone says “you know what I mean”, you can be pretty sure that *he* does not know what he means, for if he did, he would tell you.
— H. Davenport (1907–1969)

OpenMath and MathML: A shared goal

OpenMath and MathML share the goal of representing mathematics "as it is", rather than "as it ought to be". A relevant example of the difference is given by [Kamareddine & Nederpelt, 2004], where the original text is

$$\text{The function } \sqrt{|x|} \text{ is not differentiable at } 0 \tag{1}$$

while its formalised equivalent is

$$\neg(\lambda_{x:\mathbf{R}}(\sqrt{|x|}) \text{ is differentiable at } 0). \tag{2}$$

The key features are the typing of x as being in $\mathbf{R}$, and the conversion of $\sqrt{|x|}$ from an expression to a function.

OpenMath and MathML (2!): different approaches

OpenMath and MathML have rather different views of calculus:

OpenMath and MathML (2!): different approaches

OpenMath and MathML have rather different views of calculus:

- what one learned in calculus/analysis about *functions*, which *we* will write as $D_{\epsilon\delta}$: the "differentiation of ϵ–δ analysis" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\epsilon\delta} x}$, and its inverse ${}_{\epsilon\delta}\int$);

OpenMath and MathML (2!): different approaches

OpenMath and MathML have rather different views of calculus:

- what one learned in calculus/analysis about *functions*, which *we* will write as $D_{\epsilon\delta}$: the "differentiation of ϵ–δ analysis" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\epsilon\delta} x}$, and its inverse ${}_{\epsilon\delta}\int$);
- what is taught in differential algebra about (*expressions* in) differential fields, which *we* will write as D_{DA}: the "differentiation of differential algebra" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\mathrm{DA}} x}$, and its inverse ${}_{\mathrm{DA}}\int$).

OpenMath and MathML (2!): different approaches

OpenMath and MathML have rather different views of calculus:

- what one learned in calculus/analysis about *functions*, which *we* will write as $D_{\epsilon\delta}$: the "differentiation of ϵ–δ analysis" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\epsilon\delta} x}$, and its inverse ${}_{\epsilon\delta}\int$);
- what is taught in differential algebra about (*expressions* in) differential fields, which *we* will write as D_{DA}: the "differentiation of differential algebra" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\mathrm{DA}} x}$, and its inverse ${}_{\mathrm{DA}}\int$).

OpenMath and MathML (2!): different approaches

OpenMath and MathML have rather different views of calculus:

- what one learned in calculus/analysis about *functions*, which *we* will write as $D_{\epsilon\delta}$: the "differentiation of ϵ–δ analysis" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\epsilon\delta}x}$, and its inverse $_{\epsilon\delta}\int$);
- what is taught in differential algebra about (*expressions* in) differential fields, which *we* will write as D_{DA}: the "differentiation of differential algebra" (similarly $\frac{\mathrm{d}}{\mathrm{d}_{\mathrm{DA}}x}$, and its inverse $_{\mathrm{DA}}\int$).

(2) is unashamedly the former, while (1) *talks* about a function, but actually gives an expression.

This duality

shows up whenever one talks about variables: while

$$2x \neq 2y, \tag{3}$$

$$(\lambda x.2x) =, \text{ or at least } \equiv_\alpha, \ (\lambda y.2y). \tag{4}$$

This duality

shows up whenever one talks about variables: while

$$2x \neq 2y, \tag{3}$$

$$(\lambda x.2x) =, \text{ or at least } \equiv_\alpha, \ (\lambda y.2y). \tag{4}$$

So does

$$\frac{\mathrm{d}x^2}{\mathrm{d}x} = \frac{\mathrm{d}y^2}{\mathrm{d}y}? \tag{5}$$

This duality

shows up whenever one talks about variables: while

$$2x \neq 2y, \tag{3}$$

$$(\lambda x.2x) =, \text{ or at least } \equiv_\alpha, (\lambda y.2y). \tag{4}$$

So does

$$\frac{\mathrm{d}x^2}{\mathrm{d}x} = \frac{\mathrm{d}y^2}{\mathrm{d}y}? \tag{5}$$

The variables are clearly free in (3) and bound in (4).

This duality

shows up whenever one talks about variables: while

$$2x \neq 2y, \tag{3}$$

$$(\lambda x.2x) =, \text{ or at least } \equiv_\alpha, (\lambda y.2y). \tag{4}$$

So does

$$\frac{\mathrm{d}x^2}{\mathrm{d}x} = \frac{\mathrm{d}y^2}{\mathrm{d}y}? \tag{5}$$

The variables are clearly free in (3) and bound in (4). Any system which attempted to *force* either interpretation on (5) would not meet the goal stated above.

This paper

studies four areas (which in fact turn out to be inter-related):

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;
2. the `<condition>` element of MathML;

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;
2. the `<condition>` element of MathML;
3. the different handling of calculus-related operations in the two;

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;
2. the `<condition>` element of MathML;
3. the different handling of calculus-related operations in the two;
4. the "lifting" of *n*-ary operators, such as $+$ to $\sum$.

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;
2. the `<condition>` element of MathML;
3. the different handling of calculus-related operations in the two;
4. the "lifting" of *n*-ary operators, such as $+$ to $\sum$.

This paper

studies four areas (which in fact turn out to be inter-related):

1. constructions with bound variables;
2. the `<condition>` element of MathML;
3. the different handling of calculus-related operations in the two;
4. the "lifting" of *n*-ary operators, such as $+$ to $\sum$.

(We shan't talk about the last in this presentation.)

MathML 2's rules on <bvar>

MathML 2's rules on <bvar>

It doesn't really have any, but reverse engineering yields the following rule.

MathML 2's rules on <bvar>

It doesn't really have any, but reverse engineering yields the following rule.

Variables in `bvar` *constructions 'bind' the corresponding variable occurrences in the scope of the parent of the* `bvar`*. However, the variable may (e.g. $\forall$) or may not (e.g. $\frac{\mathrm{d}}{\mathrm{d}x}$) be bound in the sense of α-convertibility.*

MathML 2's rules on <bvar>

It doesn't really have any, but reverse engineering yields the following rule.
Variables in `bvar` *constructions 'bind' the corresponding variable occurrences in the scope of the parent of the* `bvar`*. However, the variable may (e.g.* $\forall$*) or may not (e.g.* $\frac{\mathrm{d}}{\mathrm{d}x}$*) be bound in the sense of* α*-convertibility.*
If there's a `<condition>`, its variables are as bound as the others.

λ-notation

λ-notation

To motivate the λ-notation, consider the everyday mathematical expression '$x - y$'. This can be thought of as defining either a function f of x or g of y ... And there is need for a notation that gives f and g different names in some systematic way. In practice *mathematicians usually avoid this need by various 'ad hoc' special notations, but these can get very clumsy when higher-order functions are involved.*
[Hindley & Seldin, 2008, p. 1]

λ-notation

> *To motivate the λ-notation, consider the everyday mathematical expression '$x - y$'. This can be thought of as defining either a function f of x or g of y ... And there is need for a notation that gives f and g different names in some systematic way.* In practice *mathematicians usually avoid this need by various 'ad hoc' special notations, but these can get very clumsy when higher-order functions are involved.*
> [Hindley & Seldin, 2008, p. 1]

MathML 3 introduces a formal `bind` to take the guessing out of the MathML 2 'rule' quoted above.

Some uses of condition are OK: e.g. $\forall x \in \mathbf{R} p(x)$

```
<apply>
  <forall/>
  <bvar><ci>x</ci></bvar>
  <condition><apply><in/><ci>x</ci><reals/></apply></condit
```

"p(x)"

```
</apply>
```

Some uses of condition are OK: e.g. $\forall x \in \mathbf{R} p(x)$

```
<apply>
  <forall/>
  <bvar><ci>x</ci></bvar>
  <condition><apply><in/><ci>x</ci><reals/></apply></condit
"p(x)"
</apply>
```

```
<OMBIND>
  <OMA>
    <OMS name="forallin" cd="quant3"/>
    <OMS name="R" cd="setname1"/>
  </OMA>
  <OMBVAR> <OMV name="x"/> </OMBVAR>
"p(x)"
</OMBIND>
```

Some uses of condition are not: e.g. $\forall x, y \in \mathbf{R} : x - y \neq 0. \frac{1}{x-y} \in \mathbf{R}$

```
<apply>
  <forall/>
  <bvar><ci>x</ci><ci>y</ci></bvar>
  <condition>
    <apply><and>
      <apply><in/><ci>x</ci><reals/></apply>
      <apply><in/><ci>y</ci><reals/></apply>
      "x\ne y"
      </apply>
  </condition>
  "\frac{1}{x-y}\in\R"
</apply>
```

$\int_0^a f(x)dx$ or $\int_{x \in D} f(x)dx$ or $\int_D f(x)dx$?

```
<lowlimit> <cn>0</cn> </lowlimit>
<uplimit> <ci>a</ci> </uplimit>
```

$\int_0^a f(x)dx$ or $\int_{x \in D} f(x)dx$ or $\int_D f(x)dx$?

```
<lowlimit> <cn>0</cn> </lowlimit>
<uplimit> <ci>a</ci> </uplimit>

<condition>
  <apply><in/>
    <ci>x</ci>
    <ci>D</ci>
  </apply>
</condition>
```

$\int_0^a f(x)dx$ or $\int_{x\in D} f(x)dx$ or $\int_D f(x)dx$?

```
<lowlimit> <cn>0</cn> </lowlimit>
<uplimit> <ci>a</ci> </uplimit>

<condition>
  <apply><in/>
    <ci>x</ci>
    <ci>D</ci>
  </apply>
</condition>

<domainofapplication>
  <ci>D</ci>
</domainofapplication>
```

$\int_0^a f(x)dx$ or $\int_{x \in D} f(x)dx$ or $\int_D f(x)dx$?

```
<lowlimit> <cn>0</cn> </lowlimit>
<uplimit> <ci>a</ci> </uplimit>

<condition>
  <apply><in/>
    <ci>x</ci>
    <ci>D</ci>
  </apply>
</condition>

<domainofapplication>
  <ci>D</ci>
</domainofapplication>
```

All equivalent in MathML 2.

$\int_0^a f(x)dx$ or $\int_{x \in D} f(x)dx$ or $\int_D f(x)dx$?

```
<lowlimit> <cn>0</cn> </lowlimit>
<uplimit> <ci>a</ci> </uplimit>

<condition>
  <apply><in/>
    <ci>x</ci>
    <ci>D</ci>
  </apply>
</condition>

<domainofapplication>
  <ci>D</ci>
</domainofapplication>
```

All equivalent in MathML 2.

OpenMath can't easily model the second.

Is this a problem? consider multi-dimensional calculus

Is this a problem? consider multi-dimensional calculus

[Borwein & Erdelyi,1995, p. 189] has a real integral over a curve in the complex plane,

$$\frac{1}{2\pi} \int_{|t|=R} \left| \frac{f(t)}{t^{n+1}} \right| |dt| \tag{6}$$

Is this a problem? consider multi-dimensional calculus

[Borwein & Erdelyi,1995, p. 189] has a real integral over a curve in the complex plane,

$$\frac{1}{2\pi}\int_{|t|=R}\left|\frac{f(t)}{t^{n+1}}\right||dt| \tag{6}$$

[Apostol, 1967, p. 413, exercise 4] has an integral where we clearly want to connect the variables in the integrand to the variables defining the set:

$$\int\int\int_{\left\{\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\leq 1\right\}}\left(\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\right)dxdydz \tag{7}$$

Solution 1: like `forallin`

```
<OMBIND>
  <OMA>
    <OMS cd="calculus_new"
         name="tripleintcond"/>
    "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
  </OMA>
  <OMBVAR>"x,y,z"</OMBVAR>
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
</OMBIND>
```

Solution 1: like `forallin`

```
<OMBIND>
  <OMA>
    <OMS cd="calculus_new"
         name="tripleintcond"/>
    "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
  </OMA>
  <OMBVAR>"x,y,z"</OMBVAR>
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
</OMBIND>
```

Forbidden since the binder is not in its own scope.

Solution 2: bind them both

```
<OMBIND>
 <OMS cd="calculus_new"
      name="tripleintcond"/>
 <OMBVAR>"x,y,z"</OMBVAR>
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
</OMBIND>
```

Solution 2: bind them both

```
<OMBIND>
 <OMS cd="calculus_new"
      name="tripleintcond"/>
 <OMBVAR>"x,y,z"</OMBVAR>
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
  "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
</OMBIND>
```

Forbidden since the binder is only allowed one argument.

Solution 3: bypass 2 artifically

```
<OMBIND>
 <OMS cd="calculus_new"
      name="tripleintcond"/>
 <OMBVAR>"x,y,z"</OMBVAR>
 <OMA>
   <OMS cd="calculus_new"
        name="tripleint_inner"/>
   "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
   "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
 </OMA>
</OMBIND>
```

Solution 3: bypass 2 artifically

```
<OMBIND>
 <OMS cd="calculus_new"
      name="tripleintcond"/>
 <OMBVAR>"x,y,z"</OMBVAR>
 <OMA>
   <OMS cd="calculus_new"
        name="tripleint_inner"/>
   "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\le1"
   "\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}"
 </OMA>
</OMBIND>
```

Legal, but unnatural.

Solution 4: bind separately

```
<OMA>
  <OMS cd="calculus_new"
       name="tripleintcond"/>
  "\lambda{x,y,z}.\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2
  "\lambda{x,y,z}.\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2
</OMA>
```

Solution 4: bind separately

```
<OMA>
  <OMS cd="calculus_new"
       name="tripleintcond"/>
  "\lambda{x,y,z}.\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2
  "\lambda{x,y,z}.\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2
</OMA>
```

which is equivalent to

```
<OMA>
  <OMS cd="calculus_new"
       name="tripleintcond"/>
  "\lambda{x,y,z}.\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2
  "\lambda{z,y,x}.\frac{z^2}{a^2}+\frac{y^2}{b^2}+\frac{x^2
</OMA>
```

Our proposal: legitimise 2

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope,

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.

2: *con*: Requires a change to the abstract description of the OpenMath standard.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

- 2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.
- 2: *con*: Requires a change to the abstract description of the OpenMath standard.
- 3: *pro*: No change to the OpenMath standard.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.

2: *con*: Requires a change to the abstract description of the OpenMath standard.

3: *pro*: No change to the OpenMath standard.

3: *con*: Needs a new, mathematically meaningless, symbol such as `tripleint_inner` for each symbol such as `tripleintcond`.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.

2: *con*: Requires a change to the abstract description of the OpenMath standard.

3: *pro*: No change to the OpenMath standard.

3: *con*: Needs a new, mathematically meaningless, symbol such as `tripleint_inner` for each symbol such as `tripleintcond`.

Our proposal: legitimise 2

Solution 1 makes bound variables have an unusual, to say the least, scope, and solution 4 is $\epsilon\delta$unnatural. The other two both achieve the fundamental goal of making both the region and the integrand subject to the *same* binding operation.

- 2: *pro*: Mathematically elegant; fits into both the XML and binary encodings of OpenMath.
- 2: *con*: Requires a change to the abstract description of the OpenMath standard.
- 3: *pro*: No change to the OpenMath standard.
- 3: *con*: Needs a new, mathematically meaningless, symbol such as `tripleint_inner` for each symbol such as `tripleintcond`.

Option 2 is our preferred route.

Our proposal rescues other cases

$\forall x, y \in \mathbf{R} : x - y \neq 0. \frac{1}{x-y} \in \mathbf{R}$ becomes

```
<OMBIND>
  <OMA>
    <OMS name="forallincond" cd="quant3"/>
    <OMS name="R" cd="setname1">
  </OMA>
  <OMBVAR><OMV name="x"/><OMV name="y"/></OMBVAR>
  "\frac{1}{x-y}\in\R"
  "x-y\ne0"
</OMBIND>
```

Conclusions

Conclusions

1. OpenMath should support both styles of calculus

Conclusions

1. OpenMath should support both styles of calculus
2. OpenMath should support a richer range of conditions, which correspond to what normal mathematicians write

Conclusions

1. OpenMath should support both styles of calculus
2. OpenMath should support a richer range of conditions, which correspond to what normal mathematicians write
3. This is most naturally done by extending `OMBIND`