Vol.2021-CE-161 No.3
2021/10/23

数式記述言語MathMLの表現形式から意味形式への変換およびオンライン小テスト作問への応用

内橋 夏実[1]　浅本 紀子[2]

概要：数式をコンピュータ上で記述する方法として XML 言語の MathML があり，表現形式と意味形式の二つの記述形式が存在する. 表現形式は数式の見た目の位置情報だけでの記述で，数式の意味構造を持たない．一方で意味形式は，厳密に数式の意味構造と演算子情報を持つ．多くのブラウザでは意味形式の表示がサポートされていないため，数式の書かれたウェブページの多くは表現形式で記述されている．しかし表現形式は意味情報を持たないため，数式処理システムなどへの応用を考えるときに直接の利用が困難である．本研究では，表現形式を意味形式に変換するツールを開発した．表現形式を内部的に厳密に記述し直す研究がされており，それと組み合わせることで，正しい意味形式に変換できる．さらに，このコンバータツールの応用の例として，数式部分が表現形式で記述されている数学の問題から，LMS である Moodle 上の STACK を利用した数学オンラインテストの問題を自動生成するツールを開発した．

キーワード：MathML，表現形式，意味形式，オンラインテスト

Conversion from Presentation Markup to Content Markup in MathML and application to create Moodle Math Quizzes

Abstract: MathML, an XML language, is a method for writing mathematical formulas on a computer. There are two methods in MathML, Presentation Markup and Content Markup. Presentation Markup is encoded by describing visual information of mathematical expressions. This has no mathematical semantic structure. On the other hand, Content Markup have the semantic structure of mathematical expressions and operator information. Many browsers do not support displaying Content Markup, so many web pages with mathematical expressions are written in Presentation Markup. However, since Presentation Markup does not have semantic information, it is difficult to use it directly when considering its application to computer algebra systems. In this research, we have developed a tool to convert Presentation Markup to Content Markup. Researched has been conducted to strictly rewrite the Presentation Markup internally, and by combining it, it can be converted into the Content Markup. Furthermore, as an example of this converter tool, we have developed a tool that automatically generates mathematical online quizzes using STACK on Moodle, which is an LMS, from mathematical questions in which mathematical expressions are described in Presentation Markup.

Keywords: MathML, Presentation Markup, Content Markup, online quiz

1. はじめに

1.1 研究背景

近年，ウェブページにおける数式表記は極めて重要となっている．その背景として，タブレット端末の普及や書籍や文書のデジタル化，LMS(Learning Management System)の推進などがある．

このような背景からウェブページにおける数式の表現方法も変化している．以前は数式を記述した画像ファイルをウェブページに埋め込むといった方法が利用されていたが，現在では数式をウェブページ上に直接記述する方法が多く利用されている．

そこで，本研究では MathML に注目した．XML 言語のひとつである MathML は，ウェブで数式を表現するため

[1] お茶の水女子大学大学院 人間文化創成科学研究科 理学専攻 情報科学コース
[2] お茶の水女子大学大学

の言語で，HTML の文書内に埋め込んで HTML 文書として扱われる．MathML には表現形式と意味形式の 2 つの記述形式が存在する．表現形式は，数式の見た目の位置情報だけでの記述で数式の意味構造を持たない．一方で意味形式は，厳密に数式の意味構造と演算子情報を持つ．多くのブラウザでは意味形式の表示がサポートされていないため，数式の書かれたウェブページの多くは表現形式で記述されている．しかし表現形式は意味情報を持たないため，数式処理システムなどへの応用を考えるときに直接の利用が困難である．

そのため，本研究では，表現形式を意味形式に変換するツールの開発を試みた．表現形式を意味形式に自動変換し数式処理可能な状態にすることで，例えば，数学教育ツールの開発など数式検索の実用化などが可能となると考えられる．さらにこのような表現形式から意味形式への変換ツールの応用の例として，数式部分が表現形式で記述されている数学の問題から，LMS である Moodle 上の STACK を利用した数学オンラインテストの問題を自動生成するツールを開発した．

1.2 関連研究

MathML の意味形式に着目した先行研究では， TEX から MathML の意味形式に変換するもの [1] や，数式情報が一部欠落した MathML 表現形式から一般的に考えられる意味を持たせ一気通貫で意味形式へ変換するもの [2] がある．本研究は後者と類似したテーマだが，欠落した数式情報をユーザの意図した通りに付加して一意で正規的な書き方に変換する研究も進められており [3][4]，本研究ではそれらと組み合わせることによって表現形式を意味形式に変換できるツールを開発した．

また，数学オンラインテスト自動生成ツールに関する関連研究としては，Word で作成した数式記述問題や代数学問題の作成に特化した問題様式の Excel ファイルを Moodle にインポート可能となる XML ファイルに変換するツールの開発 [5] がある．

2. 意味形式へのコンバーターツール

2.1 表現形式と意味形式

2.1.1 表現形式と意味形式の比較

表現形式は数式をブラウザに表示することが重要である状況，意味形式は数学的意味をコード化することが重要である状況に適したものである．

表現形式は約 30 の要素と 50 の属性で構成されており，数式の視覚的構造をコード化する．式全体は math 要素で囲まれており，この他には表 3 のような表示要素を使っている．例えば数式 x^2+1 は図 1 のように表現される．

一方で意味形式は約 140 の要素と 12 の属性で構成されており，数式の論理的意味をコード化する．要素 ci，cn は

表 1　表現形式記述の要素（一部）
Table 1　a part pf Presentation Markup elements

要素	用途	備考
mi	関数名や変数名の識別子を表す	
mn	数値を表す	
mo	演算子や区切り記号を表す	
mrow	因位の数式をグループ化する	
mfrac	分数	引数 2 つ
msqrt	2 乗根	引数 1 つ
mroot	累乗根	引数 2 つ
msub	下付き文字	x_1 など，引数 2 つ
msup	上付き文字	累乗や転置，引数 2 つ
mfenced	一対の囲い文字で本体を囲む	括弧の記述

```
<math>
    <msup>
        <mi>x</mi>
        <mn>2</mn>
    </msup>
    <mo>+</mo>
    <mn>1</mn>
</math>
```

図 1　表現形式
Fig. 1　Presentation Markup

それぞれ識別子と数値を表すのに使われる．apply 要素は，式に演算子や関数を適用するために使われる．apply 要素の第一引数は，通常は演算子や関数を表す空要素である．残りの引数は第一引数を適用する 1 つまたは複数の式である．図 2 は，x^2+1 を意味形式で表現したものである．この例では，二行目の apply 要素の第一引数は加算を表す空要素 plus，三行目は累乗を表す空要素 power である．

```
<math>
    <apply><plus/>
        <apply><power/>
            <ci>x</ci>
            <cn>2<cn>
        </apply>
        <cn>1</cn>
    </apply>
</math>
```

図 2　意味形式
Fig. 2　Content Markup

2.1.2 表現形式の見えない演算子

表現形式における見えない演算子とは，web ページで表記される際には見た目に記号として現れることはない演算子である．見えない演算子の一覧を図に示す．

表現形式では，見えない演算子が省略されていてもブラウザでは数式が見た目上は正しく表記される．例えば

表 2　見えない演算子一覧（出典:[3] の図 3.3）
Table 2　Invisible

コード番号	文字実体参照	意味	使用例
U+2061	⁡	関数	$f(x)$
U+2062	⁢	乗算	ax
U+2063	⁣	コンマ	a_{ij}
U+2064		加算	$3\frac{2}{5}$

$4x+1$ という数式の場合，ブラウザにはほぼ同じ見た目で表示されるが図のように 3 通り記述方法が存在する．A の記述方法では，乗算を表す演算子⁢ を記述することで「$4x$」が 4 と x の掛け算であることが明確に表現されている．B の記述方法では，⁢ が省略されており，「$4x$」が 4 と x の掛け算であることが明確でない．C の記述方法では，「$4x$」が 1 つの変数として扱われている．

表 3　表現形式の記述例
Table 3　Description example in Presentation Markup

A	B	C
<math>	<math>	<math>
<mn>4</mn>	<mn>4</mn>	<mi>4x</mi>
<mo>⁢</mo>	<mi>x</mi>	<mo>+</mo>
<mi>x</mi>	<mo>+</mo>	<mn>1</mn>
<mo>+</mo>	<mn>1</mn>	</math>
<mn>1</mn>	</math>	
</math>		

見えない演算子が省略された表現形式記述にユーザが意図する情報を付与して厳密に記述し直すツールは，八巻澄奈が開発した．本研究では，このツールにより明確に演算子情報が書かれた状態である表現形式記述を扱うものとする．

2.2　提案手法

意味形式のコードを出力するには，構文木が必要である．以下の順で，表現形式記述のタグを利用してリストを作成し，リストから構文木を作ることで変換を行う．

(1) 変換元ファイルの入力
(2) リストの作成
(3) リストの整理
(4) 構文木の作成・意味形式の書き出し

2.2.1　変換元ファイルの入力

与えられた HTML ファイルの MathML で記述された部分を抽出し，別のファイルにする．2.1.1 章で述べたように，MathML で記述された式全体は math 要素で囲まれている．ファイルに math 要素に囲まれた箇所が n 箇所あれば，タグ $から$までが出力された新たなファイルが n 個作成される．

2.2.2　リストの作成

MathML 表現形式記述部分が抽出されたファイルを一行ずつ読み込み，それぞれから図 3 のように要素数 9 のリストを作成する．図 3 の 5 番目から 8 番目の要素はのちに構文木を作成する際に利用するため，この時点では None にする．

index	型	
0	string	… 開始タグ　<math>,<mi>,<mo>…
1	string または list	… content （開始タグから終了タグまでで囲われた部分）
2	string	… 終了タグ　</math>,</mi>,</mo>…
3	string	… HTMLのコメントアウト
4	int	… 開始タグのみ：1，contentがあれば：2 終了タグのみ：3，コメントアウトのみ：4
5	int	… 構文木の左の子ノードのid
6	int	… 構文木の右の子ノードのid
7	int	… 構文木の親ノードのid
8	int	… id

図 3　リストの要素
Fig. 3　elements in a list

また，構文木を作成・利用する際に id から他のリストを参照できるよう，これら一行分を表したリストを要素として一行目から順に並べた 1 つのリストを作成する．このリストを list_whole と呼ぶ．

例えば図 4 のようにファイルに数式 $x^2-9=0$ が記述されていた場合，図 5 のようなリストを作成する．（図 5 は図 3 の 5 ～ 8 番目の要素は省略している．）

```
<math>
    <msup>
        <mi>x</mi>
        <mn>2</mn>
    </msup>
    <mo>-</mo>
    <mn>9</mn>
    <mo>=</mo>
    <mn>0</mn>
</math>
```

図 4　読み込むファイルの中身
Fig. 4　Contents of the file to be read

2.2.3　リストの整理

list_whole の要素の中に math 要素開始タグ以外で図 3 の 4 番目の要素が 1 であるリスト存在する場合，それに対応する 4 番目の要素が 3 であるリストを探索し，開始タグのリストから終始タグのリストまでを 1 つのリストにまとめる．図 5 の場合，図 6 のように list1 から list4 までを list1 に 1 つのリストとして統一する．（図 6 では list0 と list9，それぞれのリストの 5 ～ 8 番目の要素を省略して

list_whole ： [list0, list1, list2, ···,list9]

list0	list 1	list2	list3	list4
<math>	<msup>	<mi>	<mn>	None
None	None	x	2	None
None	None	</mi>	</mn>	</msup>
None	None	None	None	None
1	1	2	2	3

list5	list6	list7	list8	list9
<mo>	<mn>	<mo>	<mn>	None
−	9	=	0	None
</mo>	</mn>	</mo>	</mn>	</math>
None	None	None	None	None
2	2	2	2	3

図 5　生成されるリスト
Fig. 5　Generated lists

list_whole ： [list0, list1,list5, list6, list7,list8,list9]

list 1	list5	list6	list7	list8
<msup>	<mo>	<mn>	<mo>	<mn>
(list2, list3)	−	9	=	0
</msup>	</mo>	</mn>	</mo>	</mn>
None	None	None	None	None
2	2	2	2	2

x^2　　−　　9　　=　　0

図 6　整理されたリスト
Fig. 6　Organized lists

いる.）

最後に `list_whole` の要素順を id とし，8 番目の要素に id を入れる．図 6 の場合，list0，list1，list5，list6，list7，list8，list9 の id はそれぞれ 0，1，2，3，4，5，6 となる．

2.2.4　構文木の作成・意味形式の書き出し

以下の手順で構文木（図 9）を作成する．

(1)「等号・不等号，加算・減算，乗算・除算，&apply function;，項」の優先順位に従い，構文木全体の根となるリストを式の右側から探索する．

(2) 根の左側と右側で (1) を行い，左右の子ノードを決定する．左側・右側の根が決まったら，それぞれそのリストの親ノードの id を図 3 の 7 番目の要素に入れる．親ノードは，自身から見て左の子ノードにあたるリストの id を 5 番目の要素に，右の子ノードの id を 6 番目の要素に入れる．

(3) (1),(2) を根の右側と左側で分割できなくなるまで繰り返す．

図 6 の list1 のように 1 番目の要素に引数を表すリストが入っている場合は，その引数でそれぞれ (1) から (3) の手順を行い構文木を作成する．

最後に，行きがけ順に構文木の各ノードを訪れ，対応する意味形式を新しいファイルに出力する．（図 10）

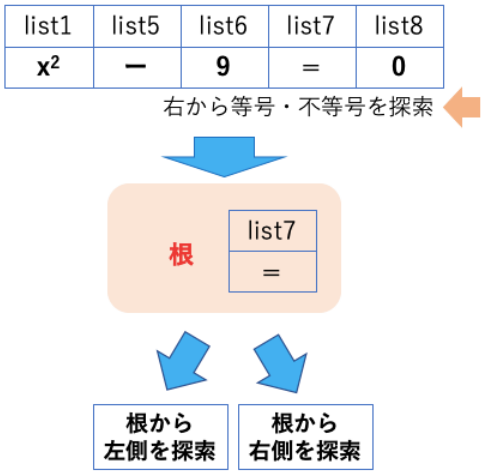

図 7　構文木の作成 1
Fig. 7　Create a syntax tree1

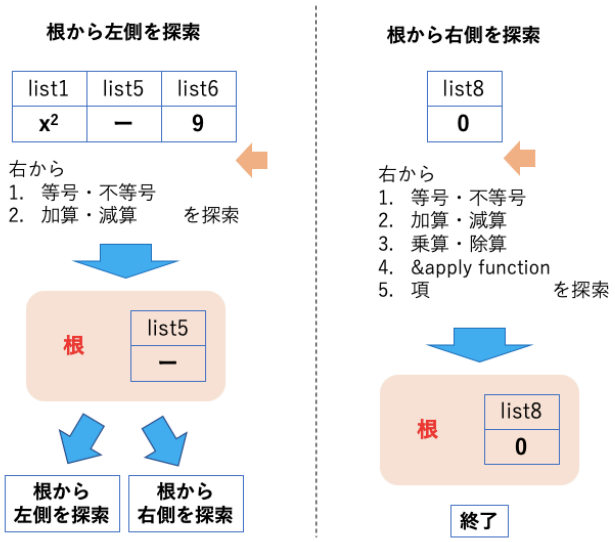

図 8　構文木の作成 2
Fig. 8　Create a syntax tre2

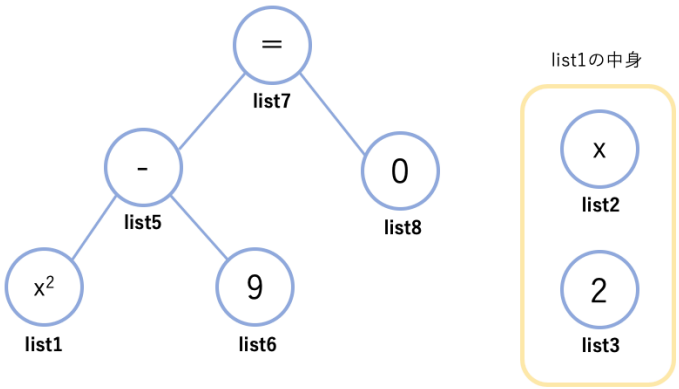

図 9　構文木
Fig. 9　Syntax tree

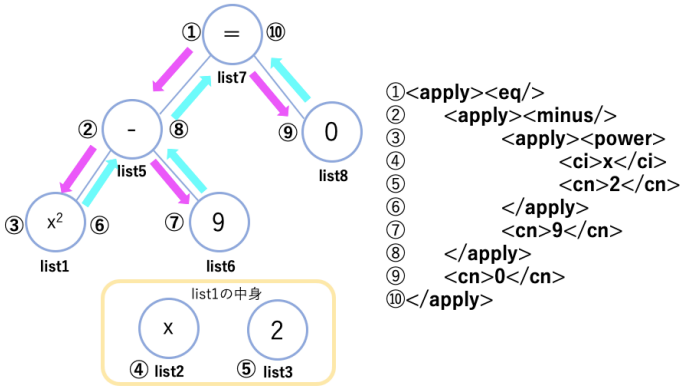

図 10　ファイルへの出力
Fig. 10　Output to file

2.3　実装結果

本ツールは「意味形式変換ツール」というディレクトリの中に変換のためのファイルが全て入っている．Python により実行されるもので，「意味形式変換ツール」ディレクトリで「makefile.py」というコマンドで，引数に意味形式に変換したいファイル名を入力するより処理が開始される．

question.html（図 11）を入力ファイルとして実行する．

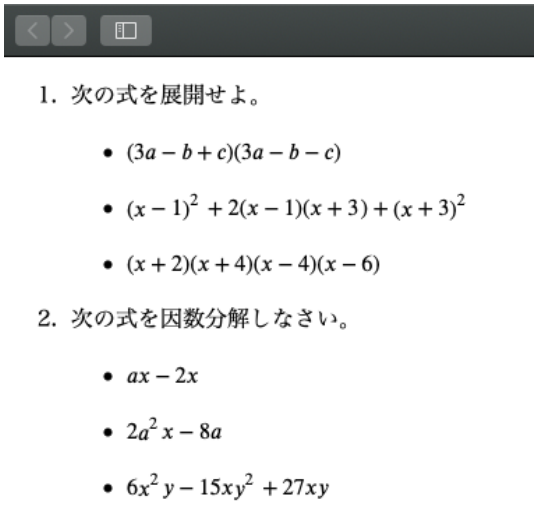

図 11　変換前のファイル：question.html（Safari）
Fig. 11　Before conversion：question.html（Safari）

question.html には MathML の開始タグ$と終了タグ$に挟まれた箇所が 6 箇所存在する．

実行すると「意味形式変換ツール」ディレクトリの中に新たに「out_folder」ディレクトリが作成され，その中に，6 箇所の MathML の表現形式で記述された部分をそれぞれ意味形式に変換したファイル「content1.xml」から「content6.xml」が出力される．

図 11 の 5 番目の数式部分を確認する．question.html では図 12 のように記述されているが，content5.xml で図 13 のように意味形式に正しく変換されたものが出力されている．

```
<math>
    <mn>2</mn>
    <mo>&InvisibleTimes;</mo>
    <msup>
        <mi>a</mi>
        <mn>2</mn>
    </msup>
    <mo>&InvisibleTimes;</mo>
    <mi>x</mi>
    <mo>−</mo>
    <mn>8</mn>
    <mo>&InvisibleTimes;</mo>
    <mi>a</mi>
</math>
```

図 12　変換前
Fig. 12　Before conversion

```
<math>
    <apply><minus/>
        <apply><times/>
            <apply><times/>
                <cn>2</cn>
                <apply><power/>
                    <ci>a</ci>
                    <cn>2</cn>
                </apply>
            </apply>
            <ci>x</ci>
        </apply>
        <apply><times/>
            <cn>8</cn>
            <ci>a</ci>
        </apply>
    </apply>
</math>
```

図 13　変換後 (content5.xml)
Fig. 13　After conversion (content5.xml)

3.　応用例

1.1 章で述べたように，MathML の表現形式を意味形式に自動変換するツールは，様々なことに応用できると考えられる．その具体的な例として著者は，図 11 のような数式部分が表現形式で記述された数学の問題を，LMS である Moodle 上の STACK を利用した数学オンラインテストの問題を自動生成するツールを開発した．

問題作成者が Moodle で小テストを作成する際，慣れるまでにある程度の時間を要する．1.2 章で紹介したように Excel や Word のフォーマットから Moodle の小テストを作成するツール [5] も開発されているが，MathML で数式が記述された Web ページから誰もが簡単にソースコード

を入手できる状況なら，問題作成者が HTML ファイルで小テストを作成し，それを Moodle の小テストに変換するという方法で問題作成者の負担がさらに軽減されるのではないかと考えた．

3.1 Moodle と STACK

Moodle とは，e ラーニングを支援する Web サービスで，生徒の学習について管理する LMS(Learning Management System) のひとつである．オンラインでの課題提出・評価，小テスト等を利用できるオープンソースの e ラーニングプラットフォームである．[6]

STACK とは，数式による回答が可能なオンラインテスト・評価システムのことである．[7] STACK での数式の入力は，基本的には Execl 等の記号と同じように，フリーの数式処理システムである Maxima の数式入力記法に従う．[8][9]

Moodle には，小テストで使用される問題をインポートおよびエクスポートするための Moodle 固有の XML フォーマットが存在する．[10] 本研究では，数式部分が MathML の表現形式で記述された数学の問題を，オンラインテスト自動生成ツールにより Moodle 用 XML ファイルに変換し，Moodle の問題バンクにインポートし正しくテストが動作するようにした．

3.2 オンラインテスト自動生成ツール

3.2.1 意味形式の利用場所

STACK によるオンラインテストの問題の例として図 14 を示す．問題文の数式部分 $(x+2)(x-6)$ は，図 15 の「問題テキスト」二行目のように，TEX で書かれている．しかし，受験者が入力した回答の正誤判定やフィードバックのために，数式処理可能な Maxima でもこの数式を「問題変数」として登録しなければならない．

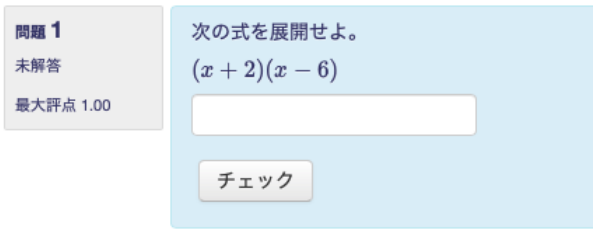

図 14 Moodle 小テスト
Fig. 14 Moodle Quiz

したがって，数学の問題が記述された HTML ファイルを Moodle 用 XML ファイルに変換する場合，MathML の表現形式で記述された数式を TEX と Maxima にそれぞれ変換しなければならない．しかし，数式の見た目の位置情報であり変数や定数，演算子の羅列である表現形式から TEX や Maxima に変換することは手間のかかる作業である．そ

図 15 小テスト作成
Fig. 15 Create a STACK Question

こで，MathML 表現形式記述を意味形式に変換し，意味形式記述から TEX と Maxima に自動で変換する．意味形式は意味構造を持つため，TEX や Maxima への変換は容易に可能となる．

3.2.2 意味形式から多言語への変換手法

図 16 のように演算子スタックと引数スタックを用意し，意味形式で数式が記述されたファイルを一行ずつ読み込んで以下の手法を繰り返す．

apply 要素開始タグ＋演算子を表す空要素の場合
演算子スタックに演算子を push する．

変数，数値の場合
引数スタックに変数または数値を push する．

apply 要素終了タグの場合
演算子スタックを pop し，その演算子がとる引数の数の分だけ引数スタックを pop する．演算子に引数を適応したものを引数スタックに push する．

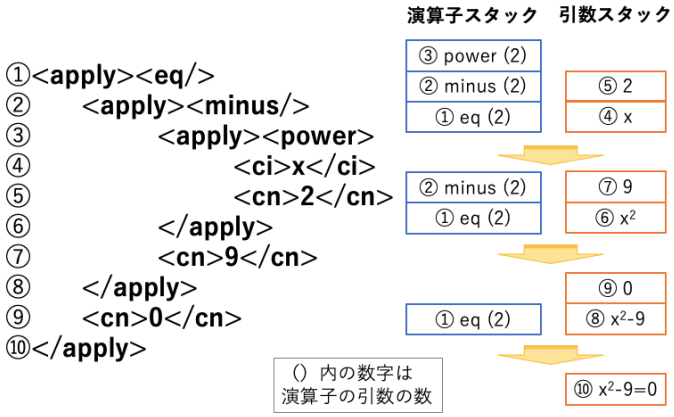

図 16 意味形式からの変換
Fig. 16 Conversion from Content Markup

3.2.3 本ツールの課題点

本ツールは，問題文に書かれた数式をどのように処理したものを回答として求めるかを解析することはできない．例えば 2.3 章の questioin.html（図 11）の 1 問目の場合，回答者に式を展開させるのか，あるいは a，b，または c で

微分するのか，様々な可能性があるが問題の文章から判断することはできず，問題作成者からの指示が必要である．

そこで，本研究では MathML の開始タグ<math>の class 属性により数式をどのように処理する問題なのかがあらかじめ分かる状態であると仮定し開発した．（表 4，図 17）

表 4　class 属性
Table 4　Class attribute

クラス属性	意味
n-order-equation_変数_次数 例）n-order-equation_x_2	2 次元方程式を x について解く
linear-eqation_変数 1，変数 2,..，変数 n 例）linear-equation_x,y	x と y の連立 一次方程式を解く
factor	因数分解する
expand	展開する
diff_変数_微分する回数 例）diff_x_1	x で 1 階微分する
integral	積分する

```
ody>
  <ol>
    <li type="1">次の式を展開せよ。</li>
    <p></p>
      <ul>
        <li type="disc">
          <math class="expand">
            <mfenced>
              <mrow>
                <mn>3</mn>
                <mo>&InvisibleTimes;</mo>
                <mi>a</mi>
                <mo>−</mo>
                <mi>b</mi>
                <mo>+</mo>
```

図 17　クラス属性利用例
Fig. 17　Usage example of class attribute

3.2.4　実行結果

本ツールを実行する手順と結果を示す．実行例はお茶の水女子大学で運用されている Moodle（Chimes2019）でおこない，画面キャプチャは全てそこからとっている．（moodle3.7.9, STACK4.2.2, maxima5.41.0）

「数学オンラインテスト自動生成ツール」ディレクトリで「makefile.py」というコマンドで実行される．引数に Moodle 用 XML ファイルに変換したい HTML ファイルを入力する．question.html（図 11）を引数として渡すと，実行後，「数学オンラインテスト自動生成ツール」ディレクトリ内に「quiz.xml」という Moodle 用 XML ファイルが作成される．

Moodle の問題バンクに quiz.xml をインポートする（図 18）と，図 19 のような画面になる．問題バンクへ移動すると，問題がインポートされていることが分かる（図 20）．図 20 の「expand1」をプレビューした様子が，図 21 である．回答が正しい場合（図 22），誤りの場合（図 23）も正しく動作していることが分かる．

図 18　インポート 1
Fig. 18　Import to Moodle1

図 19　インポート 2
Fig. 19　Import to Moodle2

図 20 インポート 3
Fig. 20 Import to Moodle3

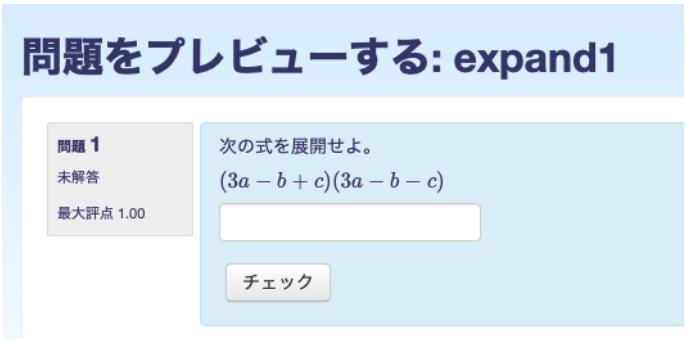

図 21 プレビュー 1
Fig. 21 Preview1

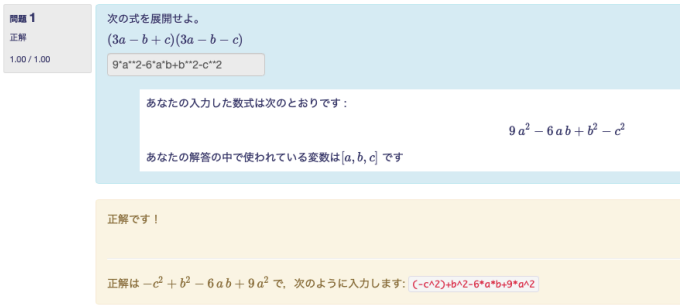

図 22 プレビュー 2
Fig. 22 Preview2

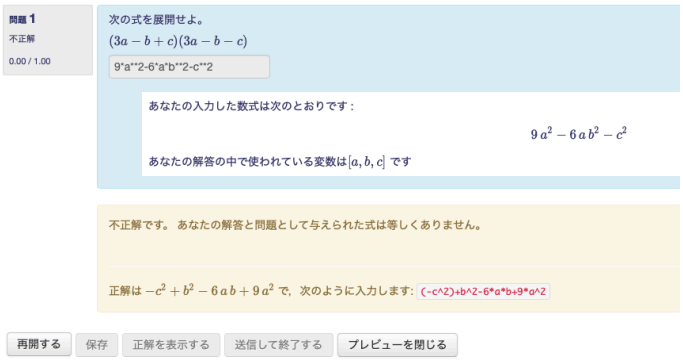

図 23 プレビュー 3
Fig. 23 Preview3

4. まとめと今後の課題

MathML の表現形式から意味形式への変換は，高校数学の数 I と数 II の問題を扱えるように，加減乗除，分数，指数，対数，平方根，累乗根，微分，積分，三角関数に対応している．高校数学で学習する範囲の数式でまだ対応できていないものは，ベクトル，虚数，シグマ演算子などが挙げられる．

オンライン小テスト自動生成ツールは，3.2.3 章で述べたように，問題文から小テストを自動生成することはできない．字句解析や機械学習を用いてそれを実現できるようにすることを目指す．

また，どちらも python3 の実行環境でターミナルコマンドを入力することによって変換結果を取得する形式になっている．ウェブアプリケーションの形で実装できるようにし，よりユーザビリティを高めることを目指す．

参考文献

[1] 渡辺千晶：視覚障害者学習支援のための MathML 変換，研究報告アクセシビリティ（AAC），2017-AAC-3.
[2] 荒川玲佳：MathML3.0 における表現形式から意味形式へのコンバーターツールの開発，研究報告アクセシビリティ（AAC），2019-AAC-11.
[3] 渡辺裕美：MathML における一意な表現形式記述への自動変換，お茶の水女子大学大学院　人間文化創成科学研究科　理学専攻　情報科学コース　修士論文 (2020).
[4] 八巻澄奈：MathML における一意な表現形式記述への変換ツールの開発，情報処理学会 コンピュータと教育研究会 161 回研究発表会 （発表予定）.
[5] 林篤，上木佐季子，遠山和大，中原敬広：Moodle 小テスト問題の一括作成 ー数式記述問題支援ツールの改良及び代数学小テスト問題作成支援ツールの開発ー，MoodleMoot Japan 2020 Proceedings.
[6] moodle： 入手先 ⟨https://moodle.org⟩
[7] STACK — Online assessment： 入手先 ⟨https://stack-assessment.org⟩
[8] 玉田瑛子：高校数学オンラインテストにおける教師支援システムの開発，お茶の水女子大学大学院　人間文化創成科学研究科　理学専攻　情報科学コース　修士論文 (2021).
[9] 中村泰之：数学 e ラーニング 数学解答評価システム STACK と Moodle による理工系教育，東京電機大学出版局 (2010).
[10] Moodle Docs： Moodle XML フォーマット 入手先 ⟨https://docs.moodle.org/3x/ja/Moodle_XML フォーマット⟩（参照 2021-09-21）.
[11] WOLFRAM：MathML を使う，入手先 ⟨https://reference.wolfram.com/language/XML/tutorial/MathML.html.ja?source=footer⟩（参照 2021-09-21）.
[12] 道廣勇司：MathML 数式組版入門 Ver1.1，アンテナハウス CAS 電子出版 (2017).
[13] MathML マニュアル，入手先 ⟨http://toshichan.be.fukui-nct.ac.jp/tsujino/mathml/chapter4.xhtml⟩（参照 2021-09-21）.