数理解析研究所講究録
第1780巻 2012年 130-140

MathML記述の数式を対象とした木構造に基づく検索システムの提案・開発

静岡大学情報学部　宮崎　佳典 (Yoshinori Miyazaki)
Faculty of Informatics,
Shizuoka University
静岡大学大学院情報学研究科　林　佳樹 (Yoshiki Hayashi)
Graduate School of Informatics,
Shizuoka University

1　はじめに

Web上で数式を表現する際に，ユーザは多くの妥協を強いられてきた．数式の持つ特徴を維持する，標準的な記法がなかったためである．これに対して近年，Web上で数式を扱う技術としてMathMLが浸透しつつある．そのMathML表記の数式に対し，検索システムがいくつか提案されてきている．ただ，それらの先行研究には，あいまい検索機能を持たない，扱える数式に大きな制約がある，ヒット率が低く実用レベルに達していない，等の点でいずれも部分的な実装にとどまっているようである．これに対し，本稿はMathMLで記述された数式を対象としたあいまい数式検索システムを提唱するものであり，MathMLの持つ木構造表現を用いたアルゴリズムを実装し，部分一致やあいまい検索が可能で，かつ高精度な数式検索システム開発を研究の目的とする．

1.1　背景

近年のWebの発達と共に，必然的に検索機能の重要性は高まっている．ことさらに，数式は数学はもちろん物理学や経済学などの多くの分野で用いられ，教育という側面からも初等教育から利用されている．数式を検索する一例として，GoogleやYahoo!等の検索エンジンを用いたキーワード検索が考えられるが，数式を検索する際に考えられるi. 定義や定理名からの検索 ii. プレーンテキストからの検索 はいずれも限定的である．というのも，iについてはすべての数式において定義や定理名があるわけではなく，それらが無い数式を検索することはできない．iiについては，プレーンテキストでは数式が複雑になればなるほど，同一の数式に対する表現法が多岐に渡る（例：$\int_1^2 f(x)dx$ は `∫ [1,2] f(x)dx` とも `∫ [1→2] f(x)dx`, `∫_1^2 f(x)dx` とも書かれることがある）等が検索への障害となる．また，構造の複雑な数式をプレーンテキストで入力するのは至難の業である．そもそも数式はWeb上では画像に変換されているケースも多く，そうなると特定の手立て[1]を講じない限り検索は困難である．

[1]例えば，IMGタグにALTオプションを付け，そこに数式情報を埋め込むことができるが，上述の問題に結局は帰着する．画像認識技術を用いることも考えられるが，精度面の問題も挙げられる．

1.2 検索システムに対するニーズ

研究に先立ち，学習者の数式検索経験や理想とする数式検索システムの機能を把握する為，大学で担当する計4講義の受講者125名[2]を対象にアンケートを実施した．

アンケートにおいて，オンライン・オフラインにかかわらず数式検索経験の有無について質問したところ，「ある」を選択した人数は102人であった．また，検索の頻度について質問した際に多かった回答を下に記述する．括弧内の数字は，各々の回答数である．

●1コマの講義中に数回(65名) ●数式を扱う問題2，3問につき1回(8名) ●テスト勉強の際に頻繁に調べる(4名)

よって学習者は学習中にある一定数，数式検索を行っている事が分かる．また，数式検索の手段について質問したところ，次の回答結果が得られた（複数回答可）．

●教科書やノートから手当たり次第に探した(79名) ●数式の定理の名前を基に索引で探した(33名) ●数式に関する用語を使ってインターネットで探した(8名)

また，数式を検索する際に，その数式をどの程度覚えていたかについては

●数式を完全に覚えていた(7名) ●数式をあいまいながら覚えていた(83名) ●定理や定義名は分かっていたが，数式自体は思いだせなかった(35名) ●その他(2名)

のような結果となった（複数回答可）．この結果から，検索の際，当該の定理名などを索引で検索するケースは少数派である点や，探そうとしている数式に対しての記憶は非常にあいまいであることが分かる．

今回アンケートを実施し，数式に接する機会の多い理系の学生のみならず，文系の学生からの意見も吸収する中で，学習者が求める数式検索システムは「数式に対して一部のあいまいな記憶からの検索が可能なシステム」であると考える事ができる．

1.3 研究の目的

本稿では，学習者がe-LearningなどのWeb上の教育システム内で数式検索システムを利用することを想定し，部分一致検索を基本の検索方法としたあいまい検索に対応する数式検索システムの構築を行うことを目的とする．それに加え，検索クエリを生成するのに手間取らないよう，我々が実際入力するような2次元表記の数式入力で検索ができるシステムにするねらいもある．検索対象とする数式はMathMLのプレゼンテーション・マークアップ（詳細は後述する）で記述されたものとする．MathMLの性質上，検索精度を上げるために木構造表現法を用いた検索アルゴリズムを提唱する．本検索システムが実現されることにより，学習者がe-Learningなどの教育システム内において，精緻なかつ融通の利く数式検索が可能となり，学習活動が継続される，関連する数式がヒットすることで周辺領域の学習意欲を誘うなど，より一層の学習効果を生むという副次的な効果も期待される．

---

[2]学部生（理系）向けの「微分積分学II」(40名)，大学院生（理系）向け「計算過程論」(12名)，学部生（文理融合系）向け「基礎数学II」(42名) [以上静岡大学]，学部生（文系）向け「情報機器の操作と応用II」(31名) [以上常葉学園大学]．

2 MathML記述データ用検索システムにかかわる先行研究

2.1 MathMLとは

MathML[1] はWWW上の数式表現の標準技術としてW3C[2]から勧告されているXMLベースの数式記述用マークアップ言語である．MathMLは現在3.0が2010年10月に勧告され，今後もより様々なシステムに取り入れられていくと考えられる．数式の表記をメタデータとして持つプレゼンテーション・マークアップ(Presentation Markup)と，数式の意味をメタデータとして持つコンテンツ・マークアップ(Content Markup)の2つのカテゴリが存在し，同時にこの両者を合わせて記述するミックスド・マークアップ(Mixed Markup)も併せ持つ．表1にプレゼンテーション・マークアップの代表的なタグの数例を示す．

MathMLはFireFoxをはじめとするMozillaベースのWebブラウザ，他にもAmayaなどのWebブラウザが標準でサポートしている．Internet Explorerでは現在標準でサポートされていないが，MathPlayer[3]（フリーソフト）をインストールすることで，MathMLで記述された数式がWebブラウザ上で自然な数式として閲覧可能となる．MathMLのソースコードを生成するソフトとして，LaTeXコードをMathMLコードに変換するコンバータソフトやMathMLコードを出力する数式作成支援ソフトなどがある．

MathMLは，Web上の数式メタデータの標準技術としてW3Cが勧告する点，さらに，LaTeXと比較して他のXMLボキャブラリと柔軟に連携できる点，(コンテンツ・マークアップの場合は）数式の意味情報を付加できる点などからWeb上において数式を表現する技術として優位性が高いと言える．LMS（Learning Management System，学習管理システム）や多くのソフトウェアで保存や読み込み形式として採用されている事実はその裏付けとなろう．上述のように，コンテンツ・マークアップは数式の意味をより確実に端的に表せるため，検索という観点ではプレゼンテーション・マークアップよりも適している．しかし，LaTeXからMathMLへのコンバータソフトと数式作成支援ソフトで出力されるMathMLの多くはプレゼンテーション・マークアップである．実際，WWW上で流通しているMathMLデータのほとんどはプレゼンテーション・マークアップのデータである．プレゼンテーション・マークアップからコンテンツ・マークアップへの変換技術も開発されているが，大武らの研究[4]にもあるように，MathMLのプレゼンテーション・マークアップの記述の自由度を完全に吸収することは難しい．

以上より，現時点ではMathMLならびにその関連技術の現状を考慮し，MathMLのプレゼンテーション・マークアップを対象とした数式検索が最適であると考える．

表1: プレゼンテーション・マークアップのタグの例

タグ	概要	タグ	概要
mn	数値を表す識別子	mtable	行列や表などを記述する
mi	変数や数学記号を表す識別子	msup	上付文字
mo	演算子を表す識別子	msub	下付文字
mrow	数式をグループ化する	mfrac	分数

2.2 先行研究・関連研究

数式検索システムにおける先行研究・関連研究には次のようなものがある．まず，橋本ら [5] による数式検索のためのインデックスに関する調査がある．MathML を考慮したインデックスを利用した検索のため，高速に検索が可能という利点がある．一方で，インデックスを厳密に定義しすぎているため，あいまい検索は行えないと推察される．

また，小田切ら [6] による MathML を用いた数式検索がある．他の研究がクエリとして数式を入力するのに対して，これは検索したい数式の形や関数をテキストベースのクエリとして検索を行う．テキストベースの検索クエリのため，数式入力用シーケンスを利用者に覚えさせるという負担が発生することが課題として挙げられる．

村方ら [7] の数式検索システムはクエリが 2 次元表記の数式として生成可能で，ユーザの負担が少ないと考える．しかし，検索方法に制約があり，あいまい検索は行えない．

宮崎ら [8] による数式検索システムは，MathML で記述された数式を対象に正規表現を用いた数式検索システムを実現したものである．“*”(0 文字以上の任意の文字列) などの正規表現を使えるため，精度面では不十分ながら[3]，あいまい検索も可能となっている．しかし，数式には 2 次元の構造があり，また MathML は XML ベースのマークアップ言語ゆえ，文章上で論理構造や意味を持たせることができる事から，検索アルゴリズムとしては正規表現に比べ木構造を考慮した手法が好ましいと考えられる．

3 MathML データの前処理

本研究では 2.1 節で理由を述べたように MathML のプレゼンテーション・マークアップで記述された数式を検索の対象としている．プレゼンテーション・マークアップは，数式の見た目に関する情報を記述することを目的とした表記方法のため，同一の数式を表す際でも様々な表記が可能である．そこで，本稿では，前処理をすることで一定のルールに基づいて MathML を整形することとする．また，より効率的な検索に向け，プレゼンテーション・マークアップの元来有する木構造表現の強固な構造化を図ることとする．

3.1 MathML データの整形

mrow タグの整形

mrow は数式の水平方向の位置的な構造を明示するタグである．例えば，下の表記 (a)，表記 (b) に示す MathML コードは，どちらも “x^{y+1}” を表し，ここに自由度が存在する．mrow タグはいくつかのタグをグルーピングするなどの目的に使用されるため，1 つのタグを mrow で囲むことに副作用は発生しない．逆に mrow タグを取り除いた場合，正常な表記とならないケースは多々考えられる．そこで mrow の記述の自由度を吸収する為に，グループ化不要な 1 つのタグであっても mrow タグを付ける処理を施した．つまり，表記 (a) のような MathML コードを表記 (b) の形に整形した．

[3]構造を加味した検索が行えないため，ユーザが意図しない数式がしばしば出力されてしまうことが確認されている．

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

表記 (a)

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

表記 (b)

予約語を含む（含まない）数式記述の整形

“sin(・)”,“abc”という数式（文字列）を見たときに，人間は前者を正弦関数，後者を変数a, b, cの積と通常認識するであろう．しかしいくつかのコンバータ（MathMLに変換する）は，前者を“<mi>s</mi><mi>i</mi><mi>n</mi>”と出力してみたり，逆に後者を“<mi>abc</mi>”としたりする[4]．これらを吸収するため，よく知られる関数名は予約語として定義し，予約語と一致する文字列はまとめてmiタグで囲み，そうでない場合は文字単位でmiタグ囲みとした[5]．

スペース・タブ・改行コード・属性値の削除

MathMLデータ中にスペースやタブ，改行コードが存在した場合，検索の阻害要因となりうる．そこでデータ登録時に事前にこれらのコードを除去する．同様に本稿ではタグの属性値も除去している（例：“<mn fontcolor="blue">”は“<mn>”に変換）．

実体参照・10進数文字参照・16進数文字参照の整形

MathMLでは数学記号やギリシャ文字は複数の表記法で記述できるようになっている．これらの複数の表記法を考慮した場合，例えば，“±”の場合3種類存在し，正規表現パターンは“&(plusmn|#(0*177|x0*B1));”となり，検索処理の負担が増大する．そこで今回は，これを16進数文字参照に統一して登録した（上の例では“±”で固定）．

3.2 MathMLデータの更なる構造化

Mathematicaの開発・販売元でもあるWolfram社がWeb上で公開している数学公式集サイト[9]や，MathMLへのコンバータソフトの出力から得られるMathMLデータ形式の多くは，下図1に示すような構造をしている．以下，模式図においては，木構造の紹介を重視するため，miやmrowなどのタグ名は表記しない．

[4]当然sinが変数s,i,nの積，abcが1変数だったりすることも考えられるが，コンバータソフトはこういったことを考慮して変換しているわけではないため，今回は一律にこのルールを適用する．

[5]予約語が別の予約語を文字列として含む場合は，与えられた文字列の中で最大文字数となる文字列の予約語をまとめてmiタグで囲んだ．例として，cosecには“cos”，“sec”，“cosec”が予約語として挙げられるが，これをc・o・secでもcos・e・cでもなく，cosecとみなした．

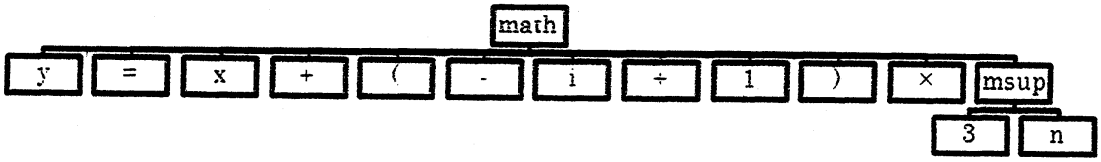

図 1: MathML の構造

図 1 に示した木構造は，数式 $y = x + (-i + 1) \times 3^n$ を MathML 表記したときのそれである．見ても分かる通り，ほとんどが並列に並んでいるだけで，構造を加味した検索を想定した際，理想的な形とは到底言えない．そこで本稿では MathML のさらなる構造化を行っている．詳細は以下に示すようになっており，演算子の優先順位が高いものから段階的に構造化を行っている（優先順位が高い順に列挙している）．なお，括弧と四則演算以外の演算子は，MathML のフォーマットとして既に構造化されている：

1) 括弧の構造化, 2) 単項演算子の構造化, 3) 二項演算子“ × ”，“ ÷ ”の構造化,
4) 二項演算子“ + ”，“ - ”の構造化, 5) “ = ”の構造化

以上の構造化によって元の（図 1 の）構造から図 2 に示すような木構造にし，数式の情報を保持した上で検索を行う．このように木構造にすることで，例えば図 3 のようなあいまい検索時の効率よい処理が見込まれる（左側のクエリ（例）に“ * ”があるため，対応する対象データ（右側）の大半をスキップすることができる様を表している）．

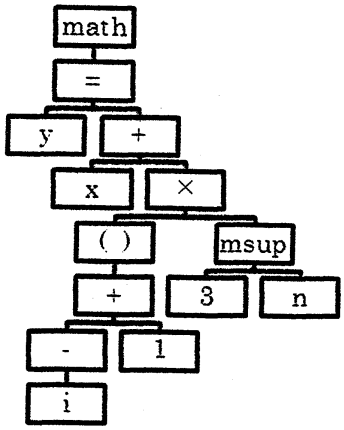

図 2: 構造化処理後の木構造

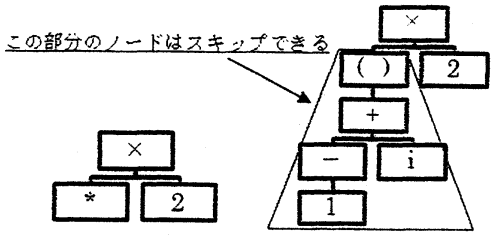

図 3: あいまい検索の構造化の利点

4 木構造に基づく検索アルゴリズム

木構造検索アルゴリズムを以下に示す．基本的な動作は (a) クエリと対象数式の先頭ノードから順に照合していく, (b)（合致すれば）ノードを各々移動し，更なる照合処理

を行う という処理の繰り返しを，両者が不一致となるか，いずれかの次のノードが枯渇するまで行うことである．また，照合を行う際にはクエリ側のノードが“*”(0文字以上の任意の文字列を表すワイルドカード）であった場合，あいまい検索処理に移行する．下記アルゴリズムの実装は，DOMに提供されているサービスを用いて比較操作等を行うため，大規模なコーディング作業を要しない．なおここに，Node(s) とはポインタsが指すノード（の値）を出力する関数，Move(s) とはポインタsを次のノード[6]に移動する関数とする：

≪基本検索アルゴリズム≫

1. クエリ，対象数式となる木の（Mathタグを除く）先頭ノードへのポインタを各々q, t, t0に割り当てる．
2. 以下を繰り返す（終了条件はNode(q)がNull値を取る，または探索失敗の場合）：
   2.1. Node(q)が“*”の場合：
   Node(q)の子ノードがない場合，q,tを各々兄弟ノード（なければ親ノードをたどって次のノード）に移動．子ノードが“*”の場合，2.1に戻って再帰処理を行う．子ノードが“*”以外の場合，qの子ノードとNode(t)が異なればMove(t)を実行，等しければMove(q)を実行する．
   2.2. Node(q)が“*”以外の場合：
   Node(q)=Node(t)の場合，Move(q), Move(t)を各々実行．そうでなければ探索失敗．
3. 2.で探索成功の場合（Node(q)=Nullの時）は終了，そうでなければqを初期値に戻し，Move(t0)を実行かつt←t0とし，t0がNull値となるまで2.を繰り返す．
4. 3.が終了した時点における探索成功/失敗の値が最終的な探索結果となる．

4.1 部分一致検索アルゴリズム

本稿では，あいまいな記憶からの数式検索を可能にする為，部分一致検索を基本としている．例えばクエリ“$x+1$”に対して，“$y+x+1$”や“y^{x+1}”をマッチする数式と見なす．本来クエリ及び検索対象の数式は，同じ仕様で整形・構造化されており，木構造としては一致するはずである．しかしながら，図4のように，クエリは“x”が“$+$”の第1子ノードであるのに対し，検索対象の数式（“$y+x+1$”）は“x”が“$+$”の第2子ノードである．そこで，今回構造化したMathMLデータから，所望の数式を取得するための法則性を抽出した．演算子（例えば“$+$”）の直左に変数または定数が現れる場合，図5左の木構造中にある○印のノードのみを調べればよいことがわかるだろう．逆に演算子の直右に変数または定数が現れる場合は図5右の○印のノードを見ればよい．

[6]優先順位は次の通りとする：1.子，2.兄弟，3.親ノードをたどって次のノード

4.2 あいまい検索アルゴリズム

原理としては，クエリ側の“*”の次のノードを参照し，そのノードと一致するものを，対象の数式中から探す処理となる．また，検索を行う中で，上記の処理（≪基本検索アルゴリズム≫）に加え数式の階層を判別する処理も行っている．数式中には上付き文字や分数といった多数の異なる水平位置のレベルが存在し，本稿ではこれを「階層」と呼ぶことにする．MathMLにおいて，この階層を変化させるタグは限られている（例えば，上付き文字であればmsup，分数であればmfracといった具合）ことを利用して不要な検索対象データの誤ヒットを防ぐのが目的である．階層判別をしないと，例えばクエリ$x^{*}+y$に対して数式“x^{1+y}”がヒットしてしまう．

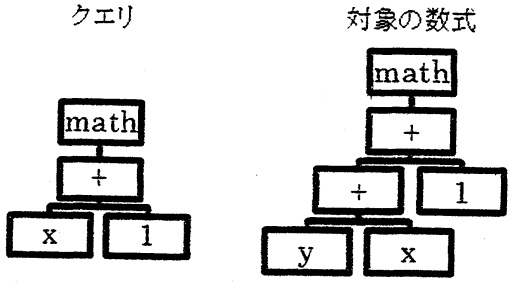

図 4: 部分一致検索における問題

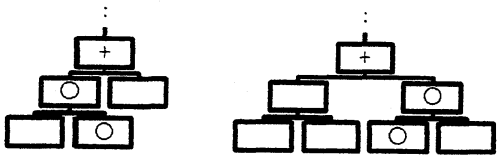

図 5: 検索時の演算子周辺の探索方法

5 実験

5.1 データセットの準備

本研究では当面の設定として，Web上の特定の教育コンテンツ内での利用を想定している．そこで，数式を最も多く扱うであろう（静岡大学情報学部の）数学系科目で教科書または参考書として指定されている計5冊のテキスト内に出現する数式の個数を調べた．その際，数式としてカウントする基準として／（不）等号・集合の包含関係を表す関係演算子が使用されているもの／段落が独立しているもの／数式番号が付加されているもの／ のいずれかとした．その結果は，(教科書1, 教科書2, 教科書3, 教科書4, 参考書1) = (2853, 1370, 2528, 2042, 1642) であった．以上の結果からいずれの個数も上回る3,000個の数式を選ぶことで実用に耐えうるものとし，今回 [9] から試行的にデータセットとして用意した．

5.2 精度比較実験（[8] をターゲットにした場合）

木構造表現を用いた検索アルゴリズムの正当性を検証するため，比較対象として正規表現法を用いた検索アルゴリズム [8] を選択した．理由は直接の先行研究に比して改良の程度を精査したかったからということと，他の先行研究で公開されているものが見当た

らなかったからである．以下に示す10つの検索クエリIR1〜IR10を数式作成支援ツール[10]を用いて発行し，正しく数式をヒットさせることができるか調査した．なお，理論値は3,000の数式に対し，目視で行った．

IR1: $\sin(z)$, IR2: $\frac{\log(*)}{*}$, IR3: $1-*y$, IR4: $\gcd(*,*)$, IR5: $\frac{\tan^2(*)}{\tan^2(*)+1}$,

IR6: $\sin(*)=\frac{\sqrt{*}}{*}$, IR7: $\sin^2(*)=\frac{*}{*}$, IR8: $\int e^*$, IR9: $\frac{*}{**}$, IR10: $m+n*-1$.

各クエリの意味は本研究では以下のように定義している：

- クエリに等号(=)がない場合は，部分一致した場合（つまり，クエリの数式表現を内部に含むような数式だった場合），
- クエリに等号がある場合は左辺は完全一致，かつ，右辺が前方一致する場合．

これは等号がある場合，右辺は例えば無限級数などで“…”を使用したり，大量の項が現れることがあるため，自然な設定と言える．なお，あいまい検索（部分一致検索）の精度を確かめたかったため，ワイルドカード“*”がIR1を除く各クエリに少なくとも1つは入っている（IR1は完全一致を確認するために用意したクエリである）．実装が完了していないため，今回はクエリに行列は含めていない．

検索結果を表2に示す．「正規表現」，「木構造表現」とあるのは，正規表現法，木構造表現法を各々実装したシステムで検索した抽出結果である．数値の中でX(Y)の形式を取る場合は，X個ヒットしたが，実際はヒットすべきではない数式がY個あった（つまり正しいヒット数はX-Y個）ことを意味する．

表2: 各クエリの2検索アルゴリズムによる検索結果ならびにその理論値

	正規表現	木構造表現	理論値		正規表現	木構造表現	理論値
IR1	128	128	128	IR6	9 (1)	8	8
IR2	23 (4)	19	19	IR7	8 (1)	7	7
IR3	3 (2)	1	1	IR8	133 (11)	122	122
IR4	15	15	15	IR9	12 (7)	0	5
IR5	1	1	1	IR10	3 (1)	0	2

表2より，両アルゴリズムの精度比較について言及してみると，まずはIR1（完全一致検索）については，両者共に正しく数式をすべて取得することに成功しているため，正常通りに動作していると考えて良いだろう．残りの9個のIR2〜IR10についてはワイルドカード“*”を含むあいまい検索となっている．先に，正規表現法の結果についてわかったことは，いずれも再現率の意味では1となっており，取りこぼしなく数式を取得することに成功していることがわかる．これは正規表現法の極めて特徴的な性質である．一方で，適合率はIR4,IR5を除いては1とはならず，多くの検索結果で本来ヒットすべきではない数式を抽出してしまっていることがわかる．適合率の幅も過半数が80-90%台であるが，50%を割り込む結果も2つ存在している（IR3,IR9）．理由は上述したように，ワイルドカード“*”がシーケンシャルな文字列に対して任意の文字列に対応

することになるため，その中に構造を含む記述があった場合に対応できていないことによる（つまり構造的に不適合な数式を峻別できていない）．

一方で，木構造表現法の結果については，IR2〜IR8と，残りのIR9,IR10に分けて考察を加える．まず前者については，適合率，再現率ともに1，そしてもちろんF値も1となる．逆に，IR9,IR10については，本手法では1件も取得できておらず，F値も0となってしまう．

いま，木構造表現法を用いた検索で正常に動作しなかった理由を，IR10を例にとって考えてみる．なお，クエリは“$m+n*-1$”であり，検索対象とする数式を“$m+n+o-1$”と設定する．当然，これはヒットしてよい数式である．それに対して，取得に失敗した理由として，両者の木構造の違いが挙げられる（図6参照）．

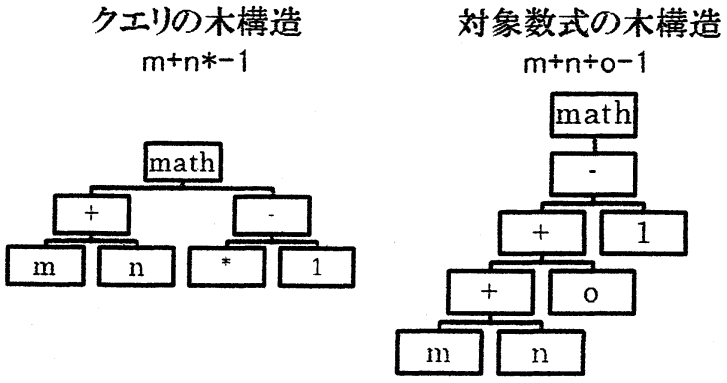

図6: 木構造検索でヒットさせられなかった数式の例

図6のように，クエリの木構造と対象の数式の木構造の構造が異なる為，ヒットできなかったのは自明である．今回MathMLコードに対して行った前処理（整形）の仕様は，演算子の階層化を密に行い，構造を厳密に生成した為，構造を問わない検索（例えば，“$m+n*-1$”というクエリでは木構造の異なる“$m+nz-1$”や“$m+n+y-1$”をヒットさせる）には十分に対応しきれていないと判断される．

それ以外にも，まだアルゴリズムの不具合については精査をする必要があろう．MathMLの自由度を吸収し，一定のルールの下でMathMLの整形を行った上で木構造表現を用いた検索を行ったことで，多くの数式を正しく取得することに成功したことは朗報と言えよう．しかし一方で，構造化されたクエリ・対象数式をマッチさせることができないのは，本来ならば構造に焦点を当てた検索がお家芸であるはずの木構造検索法の確立がまだ道半ばであることを示している．

6 結論

本稿ではWeb上のe-Learning用コンテンツ内において，MathML（プレゼンテーション・マークアップ）で記述された数式を対象とした検索システムの構築を行った．MathMLがXMLベースのマークアップ言語であるという性質に着目し，木構造表現を用いた検

索アルゴリズムを考案・実装[7]した．パイロット実験結果より，完成はしていないものの，先行研究（[8]）に比べユーザが意図する数式を正しく返すことが大方確認された．ユーザの負担軽減策として，不完全ながら2次元表記の数式入力が可能なツールを用意したことも本研究の特長として挙げておきたい．一方で，正確な検索結果を得られない例も含まれていたため，木構造の整形の仕様を改良し，検索アルゴリズムを根本から見直す必要もあろう．両検索アルゴリズムで甲乙つけがたい結果となった検索例があったことも事実である．1つの方向性として木構造表現法と正規表現法のハイブリッド化が考えられ，相補的な検索ができるか検討に値しよう．また，短期的な課題としては，現時点で実装が済んでいない行列表現等への対応，より多様なアドバンスト検索の実装（例えばNot検索や複合検索など），XQueryを基幹アルゴリズムに据えた検索システムの提案などが挙げられる．将来の展望としては，検索速度を飛躍的に向上させ，教育システムでの使用のみならず，検索エンジンとして対象範囲を広げたいと考えている．

参考文献

[1] MathML, http://www.w3.org/TR/MathML3/.

[2] W3C, http://w3.org/.

[3] MathPlayer, http://www.dessci.com/en/products/mathplayer/.

[4] 大武信之, 金堀利洋:「XML数式意味記述のための半自動変換」, FIT(情報科学技術フォーラム), 2003.

[5] 橋本英樹, 土方嘉徳, 西田正吾:「MathMLを対象とした数式検索のためのインデックスに関する調査」, 情報処理学会研究報告データベース・システム研究会報告, Vol.2007, No.54, pp.55–59, 2007.

[6] 小田切健一, 村田剛志:「MathMLを用いた数式検索」, The 22nd Annual Conference of the Japanese Society for Artificial Intelligence, 2008.

[7] 村方衛, 岸本貞弥, 大塚透, 中西崇文, 櫻井鉄也, 北川高嗣:「複合数式検索を対象とした入力支援GUI "MathGUIde"の実現」, 第17回データ工学ワークショップ(DEWS2006), 2006.

[8] Y. Miyazaki, Y. Iguchi, Development of Information-Retrieval Tool for MathML-based Math Expressions, Proceedings of the 16th International Conference on Computers in Education (ICCE2008), pp.419–426, 2008.

[9] Wolfram Functions Site, http://functions.wolfra.com/.

[10] 小笠原正彦, 宮崎佳典:「Javaアプレットを用いたブログシステム上での数式作成支援ツールの開発」, 情報処理学会第71回全国大会, pp. (1)–711–712, 2009.

[7]実装はまだ部分的であり，未実装のもの[行列など]もいくつか存在する