B-005

類似性を考慮した数式検索手法の提案
A Method of Search System for Mathematical Expressions Considering Similarities

横井 啓介† 相澤 彰子†‡
Keisuke Yokoi Akiko Aizawa

1. はじめに

Web 上に膨大な量のページが存在する現在において，検索システムは我々の生活に不可欠なものとなっている．一方で，数式というものは独特かつ複雑な構造を持っているため，自然言語を対象とした従来の検索システムではこれらの数式をうまく扱うことができず，数式を検索するためにはその構造を考慮した新しい検索が必要であると言える．

数式検索に関する先行研究の多くは，XML の一種である Mathematical Markup Language (MathML)[1] を利用している．MathML は数式の構造情報を保持した表記法であり，数式を Web 上に表現する目的に特化した Presentation Markup と，意味構造の表現を重視した Content Markup という二種類が存在する．Adeel ら[2] は MathML 式から正規表現を用いて式の特徴をキーワードとして抽出して検索に利用する手法を提案している．この手法は現在一般的な自然言語の検索システムを利用している点で利便性がある．しかしこの手法では数式の細かい意味構造を犠牲にせざるを得ない．橋本ら[3] は Presentation Markup で書かれた MathML 式の DOM 構造の XPath を用いた検索システムを実現させている．XPath を用いることで数式の構造を検索要素に入れ，形が近い数式を探すことができる．しかしながら，彼らの手法ではあてはまる数式を出力するのみで，ランキングは行わないためにユーザは結果を判断しにくい．一方で小田切ら[4] は，Content Markup で書かれた MathML 式を，独自のクエリ記法を用いて木構造マッチングを行っている．この手法では構造が正確に当てはまる数式を完全に抽出することが可能であるが，ユーザによるクエリの入力が困難で，またワイルドカードのような記法を除き，構造が正確にマッチしていないと出力されない．

そこで本研究では，MathML の Content Markup で書かれた式を用いて，類似度を評価し，ランキングして出力する類似数式検索手法を提案する．

本稿の構成は以下の通りである．まず 2 章では MathML の二種類の記法について簡単に紹介する．次に，3 章では我々の検索システムにおいて類似度を測る指標として用いる Subpath Set について述べ，続く 4 章では Subpath Set の有効性を引き立たせるための Content Markup の再構築の手法について説明する．以上の手法を用いて数式検索システムを実装し，それを用いた実験及び結果を 5 章で示す．最後に 6 章で結論と今後の課題についてまとめる．

2. MathML

MathML とは、W3C[1] による XML ベースの数式記述言語である．XML の一種であるので，Web ページに簡単に挿入することができる．MathML には，Presentation Markup と Content Markup の二種類の記法が存在する．

Presentation Markup の一番の目的は，数式を Web 上に表現することである．Presentation Markup は数式を目に見える形で構築し表現することができ，今日では Presentation Markup のみが Web 上に記述され，用いられることも多い．タグの種類は Web 上に表現するための視覚的なものばかりで，全部で 30 種類ほどである．それゆえ，類似式を検索する際には，タグの種類が少なく，構造としての意味を内部に含んでいない Presentation Markup では容易ではない．

一方，Content Markup は数式の意味構造を表現することに重点を置いている．多くの演算子や関数がそれぞれタグに割り当てられているので、タグの種類は 100 種類を超えるほどに豊富である．それゆえに Content Markup を用いることで，数式の構造を知ることが容易になる．しかし現在では Web 上に記述が存在しないことも多く，存在する場合は Presentation Markup による記述の中に，注釈として annotation-xml タグの中に付けられることがほとんどである．ただし，Mathematica[6] などの技術計算系アプリケーションを用いて Presentation Markup のみが記述された式から Content Markup 記述を作り出すことも可能である．

3. Subpath Set

類似数式検索システムにおいて，どのように数式が「似ている」かを定義することは非常に重要である．本研究では MathML で書かれた数式を扱うにあたり，数式を木構造として捉え，その木構造としての類似度を測る尺度として Subpath Set を用いた．Subpath Set は市川ら[5] によって，テキスト構文構造の類似度を測る尺度として提案されたものである．本研究ではそれを数式検索に応用した．市川らによる Subpath Set は以下のように定義されている[5].

・部分経路 (Subpath) を，「根から葉までの経路とその一部」として定義し，各々の構文木の部分経路の集合から，それらの出現頻度ベクトルに基づいて類似度を計算する．

Subpath Set の一例を図 1 に示す．左側がこの例に用いた木構造であり，右側がその木構造から作られた Subpath Set である．

† 東京大学 情報理工学系研究科 コンピュータ科学専攻

‡ 国立情報学研究所

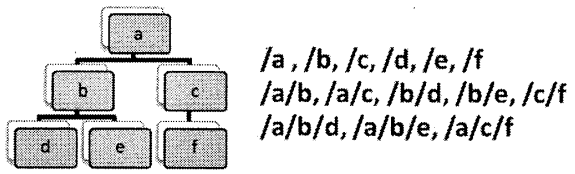

図 1　例: Subpath Set

類似度の計算方法にも様々なものがある．その中で，今回は Jaccard 係数を用いて実装を行った．Jaccard 係数の定義を (1) に示す．ここで，t_i はそれぞれ木を表し，$S(t_i)$ は木 t_i の Subpath Set を表している．

$$\frac{\|S(t_1) \cap S(t_2)\|}{\|S(t_1) \cup S(t_2)\|} \quad (1)$$

Jaccard 係数は先行実験において，我々の類似数式検索においては Dice 係数，Simpson 係数，Cosine 係数やその他の評価尺度よりも優れた結果を残していると判断したため，本論文では全て Subpath Set による，Jaccard 係数を用いた類似検索を用いている．

4. Content-Markup MathML 構造変換

Content Markup で書かれた数式において，"apply" タグは最も出現頻度の高いタグである．実験に扱ったデータにおいて，実に全タグのうちの 40～50%が "apply" タグであった．"apply" タグの意味については，W3C[1] による MathML の規格では以下のように定められている．

The most fundamental way of building up a mathematical expression in MathML content markup is the apply construct. An apply element typically applies an operator to its arguments. It corresponds to a complete mathematical expression. Roughly speaking, this means a piece of mathematics that could be surrounded by parentheses or "logical brackets" without changing its meaning.

In MathML 2.0, the apply construct is used with all operators, including logical operators.

このように"apply" タグは，その長男にあたる場所にある演算子タグを，その他の子に適用する場合に用いられる．また逆に，演算子や関数を適用する場合にはいつでも"apply" タグが存在する．この"apply" タグの存在は確かに演算子や関数の適用範囲を明らかにしているという点では有用だが，我々の Subpath Set を用いた類似数式検索においては，ノイズのある情報を増加させてしまうという問題があった．そこで我々は，この Content Markup を基にして，より Subpath Set を有効に扱うために MathML 構造に変換を加えた．具体的には以下の手法で変換を行う．

- "apply" タグの長男を親である"apply" タグの位置に置き換え，"apply" タグを取り除く．

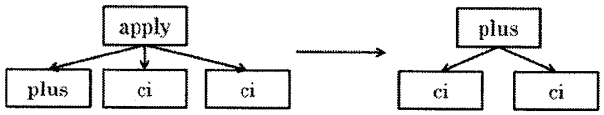

図 2　例: *apply-free* Content Markup への変換

- その他の子はそのまま置き換わったタグの子となる．

このようにして置き換わった新しい記述を，*apply-free* Content Markup として新たに定める．この *apply-free* Content Markup の例を図 2 に示す．

今回の例では，"apply" タグの長男で演算子を意味する"plus" タグが，元の"apply" に置き換わっている．これによって，「ci を plus する」といった，より深い意味持った Subpath を生成することが可能になる．

例外として，"apply" タグの長男もまた"apply" タグである数式が存在する．これは適用する関数自体が複数の単語で構成されている場合である．このような"apply" タグを"double-sequence apply" と呼ぶことにする．今回我々が実験に用いた 155,607 式のうちには，1,100 式に"double-sequence apply" が存在した．このような状況に備え，変換の手順に以下を追加する．

- "double-sequence apply" が存在する場合，子の"apply" タグの長男をその"apply" タグに置き換え，その"apply" タグを親の"apply" タグに置き換える．

図 3 にこのような場合においての変換の例を示す．

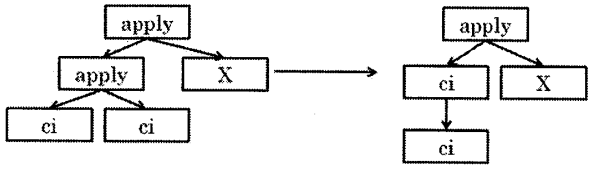

図 3　例: "double-sequence apply" の変換

5. 実験

今回の実験では，Wolfram Functions Site[7] よりクロールした 155,607 の数式を検索対象として用いた．これらの式は全て Presentation Markup 記述に加え，Annotation として Content Markup の記述も含んでいる．Content Markup で記述された式に関しては先に述べた構造変換を施し，*apply-free* Content Markup へ変換した．また実験には比較の為に Presentation Markup 記述式と，元の Content Markup 記述式も用いた．Presentation Markup 記述式に関しては，橋本ら [3] 同様，数式の意味には直接関わりがなく，数式検索を

する上で曖昧性を深めてしまう，mrow，mstyle，semantics，annotation，annotation-xml の各要素を取り除いた．

まず始めに，それぞれの 3 記法について木構造としての性質を比べ，各ノードの持つ子ノードの数と各式の木構造としての深さに関する分布をそれぞれ図 4, 図 5 に，またそれぞれの平均と最大の数値を表 1 に示す．これらから，Presentation Markup 表記においては比較的木構造が深くならず，逆に各ノードが持つ子ノードの数は多くなる傾向にあり，逆に Content Markup 表記や *apply-free* Content Markup では各ノードの持つ子ノードの数はあまり多くはならないが全体としてより深い木構造を持つ，ということがわかる．また *apply-free* Content Markup は Content Markup とほぼ同じ傾向を示すが，apply の削除によって各ノードの持つ子の数がより少なくなっていることがわかる．

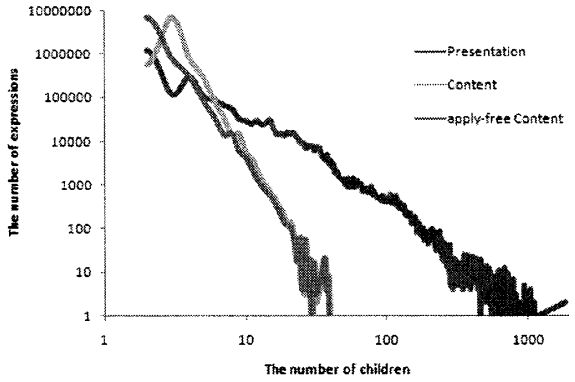

図 4　子ノードの数分布

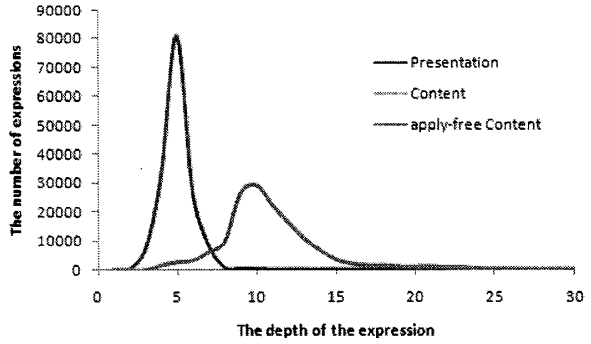

図 5　構造の深さ分布

表 1　各表記の数式構造の深さと子数

構造種類	深さ最大	深さ平均	子数最大	子数平均
Presentation	16	5.000	11,438	7.187
Content	54	11.102	77	2.903
Apply-free	54	11.102	76	2.044

本研究の実験には，これまでに述べた手法を用いて実際に実装した試作検索システムを用いる．最初に，Subpath Set の有効性，すなわち実際に類似検索としての機能を果たしていることを確認するため，適当なクエリを選び，*apply-free* Content Markup を対象として検索システムを用い，実際に結果を出力させた(表 2，表 3)．

表 2　検索例 1

クエリ：$\sin(a+b)=\sin(a)\cos(b)+\cos(a)\sin(b)$	
順位	出力数式
1	$\sin(a+b)=\sin(a)\cos(b)+\cos(a)\sin(b)$
2	$\sin(a-b)=\sin(a)\cos(b)-\cos(a)\sin(b)$
3	$\sin(a+ib)=\sin(a)\cosh(b)+i\cos(a)\sinh(b)$
4	$\cos(a-b)=\cos(a)\cos(b)+\sin(a)\sin(b)$
5	$\cos(a+b)=\cos(a)\cos(b)-\sin(a)\sin(b)$

表 3　検索例 2

クエリ：$\tan(z)=\dfrac{\sinh(iz)}{\sinh(i\pi/2+iz)}$	
順位	出力数式
1	$\tan(z)=\dfrac{\sinh(iz)}{\sinh(i\pi/2-iz)}$
2	$\tan(z)=\dfrac{\sinh(iz)}{\sinh(i\pi/2-iz)}$
3	$\sec(z)=\dfrac{i}{\sinh(i\pi/2+iz)}$
4	$\sec(z)=\dfrac{i}{\sinh(i\pi/2-iz)}$
5	$\cot(z)=\dfrac{\sinh(i\pi/2+iz)}{\sinh(iz)}$

結果より，我々の提案した手法が視覚的に「似ている」構造を持つ数式を出力できていることが確認できる．これにより，ワイルドカード等を用いることなく，構造の似ている式を得ることができることがわかった．

次に，*apply-free* Content Markup の有効性を示すため，同様の手法で Presentation Markup や Content Markup を用いて検索を行った結果と比較する．比較のために，それぞれの検索システムに対し同じクエリと，そのクエリを入力したときに「出力が期待される」数式のセットを手動で用意し，その検索結果を比較する．今回の実験に関し我々は様々なクエリと「期待」式のセットで検索を行ったが，その中でも**他の記法と順位が 3 以上異なっていた**セットを表 4 に示す．各列の P，C，A はそれぞれ Presentation Markup，Content Markup，*apply-free* Content Markup 記法を用いた検索システムによる結果を表している．また，X は上位 100 位以内に「期待」式が現れなかったことを示している．

結果を見ると，Presentation Markup による検索は類似検索に適した結果を出力していないことがわかる．これは Presentation Markup による木構造は，タグの種類が少なく，また比較的浅い構造であるため，Subpath Set に深い意味を持たせることができないためであると思われる．また今回の実験の範囲では，*apply-free* Content Markup は Content

Markup よりも良い結果を得ることができた．確かに Content Markup の方が「期待」式の順位が 1 や 2 だけ上回るセットも存在したが，3 以上上回るセットは今回の実験の範囲では現れなかった．逆に表 4 のように *apply-free* Content Markup が大きく良い結果を出すことはあった．これは Content Markup を用いた時に，有効な意味を持たない Subpath が多いために，我々の感覚的にはあまり似ていないと思われる式も上位に持ってきてしまうことがあるためだと思われる．

表 4　「期待」式の順位比較

クエリ	「期待」式	P	C	A
$\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b)$	$\sin(a-b) = \sin(a)\cos(b) - \cos(a)\sin(b)$	X	6	2
$\int \sin(z)\,dz = -\cos(z)$	$\int \sin(az)\,dz = -\frac{\cos(z)}{a}$	X	39	23
$\int z\,e^{az}dz = \frac{e^{az}(-1+az)}{a^2}$	$\int z^3\,e^{az}dz = \frac{e^{az}(-6+6az+3a^2z^2+a^3z^3)}{a^4}$	X	17	5
$\int (e^{cz})^v dz = \frac{(e^{cz})^v}{cv}$	$\int \sqrt{e^{cz}}dz = \frac{2\sqrt{e^{cz}}}{c}$	X	5	2
$\mathrm{ArcSin}(z) = \frac{3\pi}{4} - \frac{1}{2}\mathrm{ArcTan}(\frac{1-2z^2}{2z\sqrt{1-z^2}})$	$\mathrm{ArcCos}(z) = -\frac{\pi}{4} + \frac{1}{2}\mathrm{ArcTan}(\frac{1-2z^2}{2z\sqrt{1-z^2}})$	33	79	16

6. まとめ、課題

本論文では，数式検索において特に類似数式を検索するための手法を提案した．構文木の類似度を測る尺度である Subpath Set を数式構造に対して応用し，さらに *apply-free* Content Markup という Content Markup で書かれた MathML 数式の変換記法を定義し、検索に用いることで Subpath Set をより効果的に扱えるようにした．これらの方法を取ることで，より柔軟で汎用性のある数式を検索することが可能になった．

今後の課題として，まず初めに対象式の拡大が考えられる．現在は Wolfram Functions Site よりクロールした 155,607 の数式のみを対象としているが，Web 上の大量の数式を扱うといった場合には，全ての Subpath Set の類似度を測ることは困難であり，インデックスを用いるなどして候補をあらかじめ絞る等の工夫が必要となる．

第二の課題は，変数・定数をどのように扱うべきか，という問題である．この我々の提案手法では，具体的な変数や定数の中身には言及せず，すべて「変数」「定数」という同じものとして扱っている．これは検索作業を単純化し，また x と y を同じように扱えるなど汎用性を持たせることができる一方で，性能を著しく低下させる要因にも成り得る．汎用性を失わない程度にこれらを考慮する必要があると思われる．

最後に，一般的にどの数式が「似ている」かどうかを定めることは非常に難しい．現在は MathML 数式の木構造表現だけを見て類似度を定めているが，例えば同じ三角関数でも tan の数式は sin や cos の数式と形が異なる場合が多く，本手法では出力させることは難しい．このような問題に対処するには，構造だけでなく関数間の類似度を定めるなど新たな手法を考える必要がある．

参考文献

[1]. Mathematical Markup Language (MathML) Version 2.0 (Second Edition), World Wide Web Consortium. http//www.w3.org/TR/MathML2/.

[2]. Muhammad Adeel, Hui Siu Cheung, and Sikandar Hayat Khiyal: Math GO! Prototype of A Content Based Mathematical Formula Search Engine. Journal of Theoretical and Applied Information Technology, Vol 4, No 10, pp.1002-1012, 2008.

[3]. 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, 2007-DBS-142, pp.55-59, 2007.

[4]. 小田切 健一, 村田 剛志: MathML を用いた数式検索. 人工知能学会全国大会, 1F1-3, 2008.

[5]. 市川 宙, 橋本 泰一, 徳永 健伸, 田中 穂積: テキスト構文構造類似度を用いた類似文検索手法. 情報処理学会研究報告, 2005-DBS-136, pp.39-46, 2005.

[6]. Wolfram Mathematica ホームページ, Wolfram Research Inc. http://wolfram.com/products/mathematica/.

[7]. The Wolfram Functions Site, Wolfram Research Inc. http://functions.wolfram.com.