ルールベース手法による数式群の同義性判定

張純朴　加藤祥太　金上和毅　加納学
京都大学大学院情報学研究科
{zhang.chunpu.76r, katou.shouta.23v, kanegami.kazuki.55z}@st.kyoto-u.ac.jp
manabu@human.sys.i.kyoto-u.ac.jp

1　はじめに

化学や鉄鋼などのプロセス産業では，激しい競争や厳しい環境規制，急速に変化する経済状況に応じてプロセスの運転を改善していくことが求められている．しかし，現代のプロセスはより複雑になっているため，そのような要求に応えることは困難になっている [1]．プロセスでは実機を用いて試行錯誤的に実験を繰り返すことは難しいため，プロセスシミュレータが広く活用されている．プロセスシミュレータは，対象プロセスの第一原理モデル（物理モデル）を用いて，指定した条件下で装置内部の状態がどのようになるかを計算する．このため，現実の現象を適切に表現できる第一原理モデルを用意することが重要である．しかし，第一原理モデルの構築には文献調査を含めた多大なコストがかかる．本研究の最終目的は，複数の文献から情報を抽出して第一原理モデルを自動で構築する人工知能（AI）を開発することである．

第一原理モデルを構築するには変数，数式，実験データといった情報が必要である．文献によって変数の記号が異なったり数式の記述の仕方が異なったりするので，AI は文献中の変数と数式の意味を正確に判別できなければならない．

本研究では，同じ意味の変数は同じ記号で表現されていると仮定し，複数の数式で構成される数式群の同義性判定手法を提案する．なお，同一の計算をしている２つの数式群を同義であると定義する．さらに，２つの数式群の同義性を判定するウェブアプリケーションを開発したので，それを簡単に紹介する．

2　関連研究とその問題点

単語や文章を対象としている自然言語処理関連の研究に対して，数式を扱っている研究は少ない．情報検索の分野では，数式が有する階層的な情報を扱うために，数式を Symbol Layout Tree (SLT) や Operation Tree (OPT) [2][3] に変換することが有効とされている．例として，等式 $x - y^2 = 0$ の SLT と OPT を図 1 に示す．Zhong ら [4] は OPT を用いて，数式間の類似度を定義し，数式検索システムを開発した．Mansouri ら [5] は，SLT と OPT を用いて，数式をベクトル空間へ埋め込む手法を開発した．これらの研究は，数式の類似度を定義しているが，数式の見た目を重視し，数式が行っている計算を無視する傾向がある．実際，Mansouri らが開発した数式検索システム Tangent-CFT [5] を用いて数式の類似度を計算したところ，数式 $a + b = 0$ は，同一の計算をしている数式 $a = -b$ より，異なる計算をしている数式 $a - b = 0$ との類似度が高いと評価された．

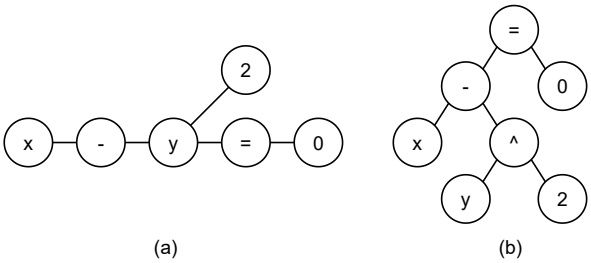

図 1: 等式 $x - y^2 = 0$ の (a) SLT と (b) OPT

数式の同義性判定に関する研究はまだないが，数式間の類似度を計算し，類似度が閾値以上であればそれらの数式を同義であると判定する方法が有効と考えられる．しかし，上述のように，数式検索で用いられている見た目重視の類似度は役に立たない．このため，数式の計算内容に注目し，数式の見た目に左右されない同義性判定手法の開発が必要である．

3　提案手法

本研究では，数式の見た目の類似度ではなく，数式の本質的な計算に注目した同義性判定手法の開発を目指す．数式を扱う先行研究では，

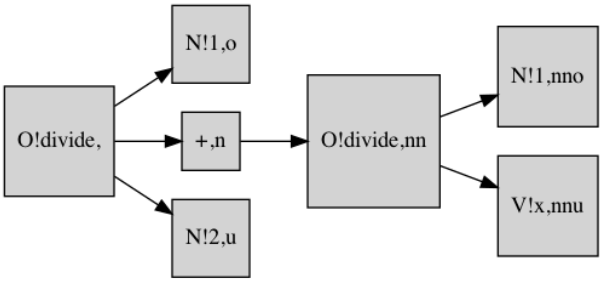

図 2: 数式 $\frac{1}{2}+\frac{1}{x}$ の SLT-tangents

Mathematical Markup Language (MathML) が用いられており，MathML はウェブページ上で公開されている文献に含まれる数式の標準記法であるため [6]，本研究では MathML 形式の数式を対象とする．Python には，SymPy [7] という代数演算を行うためのライブラリがあり，四則演算や一部の初等関数の演算を実行できる．SymPy で演算可能な数式を本稿では SymPy の数式オブジェクトと呼ぶ．本研究では MathML 形式の数式を SLT に変換し，SLT を解析することで，数式に含まれる変数および変数間の関係の情報を抽出する．これらの情報をもとに，数式の SLT を SymPy の数式オブジェクトに変換する．

3.1 数式の木構造への変換

本研究では，MathML 形式の数式を SLT に変換する際，Tangent-s formula search engine [8] の Layout-Symbol クラスの parse_from_mathml 関数を用いる．以下では，この関数によって生成された SLT を SLT-tangents と呼ぶ．数式 $\frac{1}{2}+\frac{1}{x}$ の SLT-tangents を図 2 に示す．SLT-tangents では，数式の記号や演算子が左から順に各ノードに格納される．V!，N!，O! はノードが変数，数字，演算子であることを示す．更に，子ノードと親ノードの位置関係を並べた文字列も付属の情報としてノードの中に入れられる．子ノードと親ノードの位置関係には，a (above)，b (below)，c (pre-above)，d (pre-below)，e (element)，n (next)，o (over)，u (under)，w (within) の 9 種類がある．SLT-tangents は枝分かれする場合があり，分枝箇所では子ノードと親ノードの位置関係が n 以外のものとなる．

3.2 SymPy の数式オブジェクトへの変換

数式を SymPy の数式オブジェクトへ変換するためには，数式に含まれる変数および変数間の関係という 2 つの情報を抽出する必要がある．変数については，V!タグが付与されたものを変数として抽出する．変数間の関係はノード同士の位置関係によって決まっており，その位置関係に応じて SLT-tangents を解析することで，変数間の関係の情報を持つデータを作成する．分数，指数，括弧，初等関数の変数間の関係を表 1 に示す．例えば，数式 $\frac{a+b}{c+d}$ の変数間の関係は [SLT-tangents 1, SLT-tangents 2, 'frac'] のようなデータとして保持される．保持されたデータの第一成分と第二成分はそれぞれ数式 $a+b$ と $c+d$ の SLT-tangents を指している．さらに，数式 $a+b$ の変数間関係は ['a', '+', 'b'] のようなデータとして保持される．四則演算と初等関数からなる数式は，再帰的に SLT-tangents を解析することにより，SymPy の数式オブジェクトに変換することができる．

表 1: 分数，指数，括弧，初等関数の変数間関係

計算の種類	保持されるデータ
分数	[分子, 分母, 'frac']
指数	[底, 指数, 'exp']
括弧	[括弧の中身, None, 'within']
正弦関数	[関数が作用する数式, None, 'sin']
対数関数	[底, 真数, 'log']

3.3 数式群の同義性判定

本研究では，2 つの数式群の同義性を判定することが目的であるため，単体の数式のみならず，複数の数式から構成される数式群も対象としている．図 3 に化学反応器の物質収支式を示す．3a と 3b は同一の物質収支式を表している．本稿において，ある数式群に含まれる等式が共有する変数を群内共有変数と定義する．3a の場合，変数 k，β，F_{AO}，F_{A}，r_{A} が群内共有変数である．群内共有変数の数は文献によって異なる可能性があるため，数式群の同義性判定を行う際，群内共有変数はない状態が望ましい．

提案手法では，まず数式群の自由度を比較する．数式群の自由度は一般的に，数式群に含まれる変数の数から数式の数を引いた値で定義される [1]．同一の計算をしている連立方程式ならば，自由度は等しい．その後，群内共有変数を消去する．群内共有変数を消去する処理は，変数を一個消去するたびに，数式の数も一個減るため，数式群の自由度を変えない可逆的な処理である．2 つの数式群の片方に含まれる任意の等式について，もう片方の数式群にそれと同一の計算をしている等式が存在する場合，この 2 つの数式群は同義であると判定する．以上の処理を Algorithm 1 に示す．なお，2 つの等式の同

$$\frac{dn_{\mathrm{A}}}{dt} = F_{\mathrm{AO}} - F_{\mathrm{A}} + r_{\mathrm{A}}\bar{V} \tag{1}$$

$$k = k_{\mathrm{o}} \exp\left(-\frac{\beta}{T}\right) \tag{2}$$

$$F_{\mathrm{AO}} = vC_{\mathrm{AO}} \tag{3}$$

$$F_{\mathrm{A}} = vC_{\mathrm{A}} \tag{4}$$

$$\beta = \frac{E}{R} \tag{5}$$

$$-r_{\mathrm{A}} = k \tag{6}$$

(a) 6 つの等式で書かれた物質収支式

$$\frac{dn_{\mathrm{A}}}{dt} = vC_{\mathrm{AO}} - vC_{\mathrm{A}} - k\bar{V} \tag{7}$$

$$k = k_{\mathrm{o}} \exp\left(-\frac{E}{RT}\right) \tag{8}$$

(b) 2 つの等式で書かれた物質収支式

図 3: 等式の数が異なる同義の数式群

義性判定では，その 2 つの等式が共有する任意の一つの変数について，それぞれの等式における解を求め，同一の解を得た場合，その 2 つの等式が同義であると判定する．

4 実験

4.1 問題設定

実験では，Tangent-CFT [5] と提案手法による結果を比較した．Tangent-CFT は数式の類似度しか計算できない．そこで，数式群の類似度 $S^*_{\mathrm{A,B}}$ を式 (9) で定義する．

$$S^*_{\mathrm{A,\,B}} = \frac{\sum_{j=1}^{b}\sum_{i=1}^{a} S_{i,j}}{ab} \tag{9}$$

ここで，a, b は数式群 A, B に含まれる数式の数，$S_{i,j}$ は数式群 A の i 番目の数式と数式群 B の j 番目の数式の類似度である．数式群の類似度が 0.95 以上のとき，2 つの数式群を同義であると判定する．

4.2 数式群の同義性判定結果

Tangent-CFT と提案手法で 6 組の数式または数式群の同義性を判定した結果を表 2 に示す．表 2 から分かるように，Tangent-CFT はたとえ数式群が異なる計算をしていても類似度を高く評価する場合がある．また．数式群が同一の計算をしているにも関わらず，同義と判定されるほど類似度が高くない場合もある．これは，Tangent-CFT は数式が表現する本質的な計算を無視する傾向があるためである．それに対して，提案手法は計算の規則に基づいて，同義の数式群が異なる書き方をされても正確に同義性を判定することができる．

Algorithm 1 Equation group equivalence judgement

1: E_{A} = equation group A, E_{B} = equation group B
2: f_{A} = degree of freedom of E_{A}, f_{B} = degree of freedom of E_{B}
3: **if** $f_{\mathrm{A}} \neq f_{\mathrm{B}}$ **then**
4: **return** False
5: **else**
6: $V_{\mathrm{A}} = \{v_{\mathrm{A1}}, v_{\mathrm{A2}}, \cdots, v_{\mathrm{A}N}\}$, V_{A} = variables in E_{A}
7: **for** $i = 1, 2, \cdots, N$ **do**
8: $E_{v_{\mathrm{A}i}}$ = equation group containing $v_{\mathrm{A}i}$
9: E_{new} = new equation group where $v_{\mathrm{A}i}$ is eliminated
10: remove $E_{v_{\mathrm{A}i}}$ from E_{A} and add E_{new} to E_{A}
11: **end for**
12: process E_{B} following the same procedure as line 6-10
13: **if** $\forall\, e_{\mathrm{A}} \in E_{\mathrm{A}}, \exists\, e_{\mathrm{B}} \in E_{\mathrm{B}}, e_{\mathrm{A}} == e_{\mathrm{B}}$ **then**
14: **return** True
15: **else**
16: **return** False
17: **end if**
18: **end if**

さらに，図 4 に示すように，提案手法を用いて数式群の同義性判定を行うウェブアプリケーションを作成した．このウェブアプリケーションは 2 つの数式群の同義性を判定し，結果を画面に表示させる．これを用いることで数式群の同義性を簡単に判定できる．

5 おわりに

本研究では，第一原理モデル自動構築に必要な要素技術として，数式群の同義性を判定する手法を提案した．提案手法は，MathML 形式の数式を演算可能な数式に変換することで，計算の規則に基づいて数式群の同義性を判定する．

実験により，提案手法は，有理式と初等関数を組

表 2: 提案手法と Tangent-CFT の同義性判定結果の比較

数式 1	数式 2	類似度 $S^*_{\mathbf{A},\mathbf{B}}$	判定結果 [1] Tangent-CFT	判定結果 [1] 提案手法	同義性 [2]
$a = b$	$a - b = 0$	0.876	F	T	○
$e^{\frac{b}{a}} = c$	$\frac{b}{a} = \log c$	0.949	F	T	○
$e^{\frac{a}{b}} = c$	$\frac{b}{a} = \log c$	0.949	F	F	×
$V_i = L_i - W$ $V_i y_{m+1} = L_i x_m - W x_W$	$y_{m+1} = \frac{L_i}{L_i - W} x_m - \frac{W}{L_i - W} x$	0.940	F	T	○
$V_i = L_i - W$ $V_i y_{m+1} = L_i x_m - W x_W$ $L_i = Fq + L$	$y_{m+1} = -\frac{W x_W}{Fq+L-W} + \frac{x_m (Fq+L)}{Fq+L-W}$	0.888	F	T	○
$E + R = M$ $E y_i + R x_i = M_z$	$E = \frac{M(-x_i+z_i)}{-x_i+y_i}$ $R = -E + M$	0.856	F	T	○

[1] T は同義である，F は同義ではないと判定した．
[2] ○ は同義である，× は同義ではないことを表す．

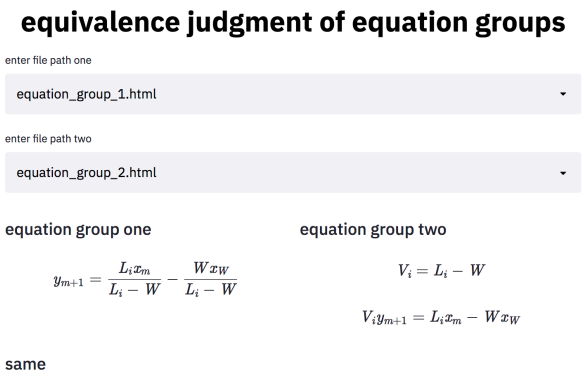

図 4: 数式群同義性判定ウェブアプリケーション

み合わせた数式群の同義性を正確に判定できることを確認した．しかし，提案手法は微分，偏微分，総和，累乗など初等関数に含まれない記号がある数式を扱うことができない．特に，微分は第一原理モデルの記述によく用いられるため，今後の研究でより多くの種類の数式に対応することを目指す．

参考文献

[1]D. Seborg, D. Mellichamp, T. Edgar, and F. Doyle III. *Process dynamics and control*. John Wiley & Sons, 2010.

[2]R. Zanibbi and D. Blostein. Recognition and retrieval of mathematical expressions. *International Journal on Document Analysis and Recognition (IJDAR)*, Vol. 15, No. 4, pp. 331–357, 2012.

[3]M. Schubotz, N. Meuschke, T. Hepp, H. Cohl, and B. Gipp. Vmext: a visualization tool for mathematical expression trees. In *International Conference on Intelligent Computer Mathematics*, pp. 340–355. Springer, 2017.

[4]W. Zhong and R. Zanibbi. Structural similarity search for formulas using leaf-root paths in operator subtrees. In *European Conference on Information Retrieval*, pp. 116–129. Springer, 2019.

[5]B. Mansouri, S. Rohatgi, D. Oard, J. Wu, C. Giles, and R. Zanibbi. Tangent-cft: An embedding model for mathematical formulas. In *Proceedings of the 2019 ACM SIGIR international conference on theory of information retrieval*, pp. 11–18, 2019.

[6]横井啓介, 松林優一郎, 相澤彰子. D-011 数式とその周辺情報を利用した数式概念検索の実現. 情報科学技術フォーラム講演論文集, Vol. 9, No. 2, pp. 117–120, aug 2010.

[7]A. Meurer, C. Smith, M. Paprocki, O. Čertík, S. Kirpichev, M. Rocklin, A. Kumar, S. Ivanov, J. Moore, S. Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, Vol. 3, p. e103, 2017.

[8]K. Davila and R. Zanibbi. Layout and semantics: Combining representations for mathematical formula search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1165–1168, 2017.